Adaptive Meta-Learning for Robust Deepfake Detection: A Multi-Agent Framework to Data Drift and Model Generalization

Dinesh Srivasthav Puvvada

2022PAI9055

A Dissertation Submitted to

Indian Institute of Technology Jammu

In Partial Fulfilment of the Requirements for the Degree of

Master of Technology in

Artificial Intelligence and Machine Learning

भारतीय प्रौद्योगिकी संस्थान जम्मू

INDIAN INSTITUTE OF TECHNOLOGY JAMMU

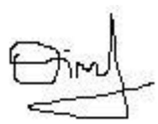
विद्याधनं सर्वधन प्रधानम्

Department of Continuing Education

October, 2024

Declaration

This thesis is submitted as a requirement for the award of Master of Technology degree in the specialization of Artificial Intelligence and Machine Learning in the Department of Continuing Education. I declare that this written submission represents my ideas in my own words, and where others' ideas or words have been included, I have adequately cited and referenced the original sources. The research embodied in this thesis has not been submitted in part or full, to any other University or Institute for the award of any other degree or diploma before. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.



(Signature)

(Dinesh Srivasthav Puvvada)

(2022PAI9055)

Approval Sheet

This thesis titled Adaptive Meta-Learning for Robust Deepfake Detection: A Multi-Agent Framework to Data Drift and Model Generalization by Dinesh Srivasthav Puvvada is approved for the degree of Master of Technology in Artificial Intelligence and Machine Learning from IIT Jammu.

Date: 05-10-2024

Signature T. Veerakumar

Dr. T. Veerakumar
Department of Electronics and Communication Engineering,
National Institute of Technology Goa
External Examiner

Signature________________________

Dr. Yamuna P Shukla
Department of Computer Science and Engineering,
Indian Institute of Technology Jammu
Internal Examiner

Signature Badri

Dr. Badri Narayan Subudhi
Department of Electrical Engineering,
Indian Institute of Technology Jammu
Supervisor

Signature Karan

Dr. Karan Nathwani
Indian Institute of Technology Jammu,
Chairman

Acknowledgements

I would like to take this opportunity to express my gratitude and sincerely thank my guide, Dr. Badri Narayan Subudhi, who helped me with various aspects of conducting research and writing this thesis. Dr. Badri permitted me to work at my own pace and explore the terrain, which helped me find confidence and gain interest in this research field. It truly helped me understand and experiment with various directions of work, several aspects of the problem statement, and get a good exposure and hands-on with several technologies, frameworks, and branches of AI. His insights and resourcefulness have helped a lot in completing my thesis work within the time frame. I acknowledge, appreciate, and value his guidance, patience, and support throughout this work.

Dedicated to the betterment of society and the advancement of research in the field of AI.

Abstract

Pioneering advancements in artificial intelligence, particularly in the space of generative AI, have given tremendous possibilities of content creation. However, this also has a down side of creating widespread misinformation, and false content. The growing sophistication and realism of deepfakes is raising several concerns around trustworthiness, privacy invasion, identify theft, primarily among others, and multifaceted adverse impact on the society, individuals, and businesses with reputational damage, and financial loss to begin with.

Many deepfake detectors have been developed to tackle this problem. Nevertheless, as for every AI or ML model, the deepfake detectors face the wrath of lack of considerable generalization to unseen scenarios, and cross-generation settings. Besides, adversarial robustness is another critical challenge where the detector drastically underperforms with the slightest imperceptible change made. Moreover, most of the state-of-the art deepfake detectors are only trained on static datasets, and have no means to continuously cope up and enhance their detection capabilities to new attack trends that emerge almost on a daily basis. These three crucial challenges though hold paramount importance for reliability in practise particularly in the deepfakes domain, are also the problems with any other AI model.

This thesis proposes an adversarial meta-learning algorithm amalgamating the prospects of task-specific adaptive sample synthesis, and consistency regularization, in a few-shot computationally non-intensive setting, with a refinement phase, thereby, focusing on both the strengths and weaknesses of the classifier, and trains it accordingly to boost the robustness and generalization of the model. It also presents a hierarchical multi-agent retrieval-augmented generation (RAG) workflow with a sample synthesis module for collecting and generating custom synthetic deepfake samples for dynamic model training adapting to the data drift. The thesis also presents a holistic framework integrating the proposed meta-learning algorithm with the hierarchical multi-agent RAG workflow for an end-to-end deepfake detection architecture together enhancing generalization, robustness, and adaptability.

This thesis establishes the problem of generalization by experimenting with 22 different models ranging from custom convolutional neural networks to pretrained models to transformers across different settings and demonstrates their drastic drop in performance across various datasets. It presents the results of the meta model trained using the proposed algorithm and demonstrates its consistent performance on various test sets outperforming the models in comparison.

Contents

List of Figures

List of Tables

List of Abbreviations

AI: Artificial intelligence

ML: Machine learning

CNN: Convolutional neural network

GAN: Generative adversarial network

RAG: Retrieval-augmented generation

LVM: Language vision model

ViT: Vision transformer

XCiT: Cross-covariance image transformer

DEiT: Data-efficient image transformer

CoaT: Co-scale conv-attentional image transformer

RECCE: reconstruction-classification learning

ACC: Accuracy

AUC: Area under the curve

Mlops: Machine learning operations

CI/CD: Continuous integration and continuous deployment

URL: Uniform resource locator

SAM: Segment anything model

SD: Stable diffusion

List of Symbols

T_i : Task-i

E_{meta} : Number of meta-epochs

D: Meta-dataset

$M_{adaptive}$: Metric to rank samples in the order of Wrong predictions with large margin > Wrong predictions with small margin > Right predictions with small margin > Right predictions with large margin

y: Sample

p_y : Prediction probability of true class of sample

$H(p)$: Entropy

$i_{misclassified}$: misclassification sign

$Margin$: Prediction margin

p_s : Highest prediction probability among incorrect classes

M_{adv} : Metric to rank samples in the reverse order of $M_{adaptive}$

L_{TSAC} : Task-specific adaptive consistency loss

$W_t^{(i)}$: Weight associated with task *t* for the sample *i*

$f(x_i),\ f(\widehat{x_i})$: Feature representations of the original and synthetic samples

y_i : Similarity sign

m : Margin parameter

L_{ADV} : Adversarial loss

$W_{adv}^{(i)}$: Adversarial task weight for sample *i*

$f(x_i),\ f\left(x_i^{adv}\right)$: Feature representations of the original and adversarial samples

N : Number of task samples

$L_{unified}$: Unified loss

L_{base} : Inner loss of Reptile algorithm

λ_1, λ_2 : Hyperparameters

$\mathrm{W_t}, \mathrm{W_{adv}}$: Adaptive weights

η : Leaning rate

1. Introduction

1.1 Overview of deepfake and types

Deepfakes have become one of the most concerning possibilities of artificial intelligence. They refer to highly realistic falsification of information or media corresponding to one or more modalities such as image, video, audio, and text, with the help of advanced deep learning and generative models. Though media manipulations, especially digital image modifications such as image splicing, colorization, use of filters, image superimposition, and so on were existing from several decades, the intent was usually or mostly for content enhancement. The term 'deepfake' was coined in 2017[1] referring to the creation of convincingly realistic fake content post a Reddit user created celebrity face swaps. Despite its malicious beginnings, deepfakes initially, gained attention for their use in entertainment, art, and harmless fun. However, their potential for misuse quickly became apparent, as deepfakes were increasingly used for various malicious purposes, with the rapid evolution of deepfake technology.

Several varieties of deepfakes are in play, each leveraging different techniques to create convincing fake content across various media formats. Some prominent examples are listed in Table *1*.

Table 1: Types of Deepfakes

Image deepfake	Video deepfake	Audio deepfake	Text deepfake	Multimodal deepfake
• Face Swap • Expression Swap • Attribute Manipulation • Face Synthesis • Face Reenactment • Lip-Syncing • Body Manipulation/swap • Background Manipulation • Background people manipulation • Foreground and background manipulation & the variants • Style Transfer • Gender manipulation	• All image deepfake methods applied to videos • Lip-Syncing • Puppet Master • Whole-Head Synthesis (Head Puppetry) • Whole-Body Synthesis (Full body puppetry)	• Text-to-Speech • Voice Conversion • Speech Synthesis • Voice Cloning or Impersonation • Audio Reenactment • Audio Style Transfer • Audio Manipulation • Audio Splicing • Audio Dubbing • Audio Gender Change	• Writing style impersonation • Synthetic text / text-to-text synthesis • Content manipulation/ falsification/ fabrication	• Audio-visual Synthesis (combining audio and video deepfakes and variants) • Text to image synthesis • Text to video synthesis • Text to audio synthesis

[1] https://mitsloan.mit.edu/ideas-made-to-matter/deepfakes-explained

• Age Progression/ Regression				

1.2 Impact of deepfakes and the growing concern

More advanced deepfakes combine two or more modalities to create multimodal deepfake content. For instance, a single video could feature a synthesized face, a manipulated background, a cloned voice, and modified lip-syncing, all working together to create an extremely convincing false narrative. The complexity and realism of such multimodal deepfakes make them extremely challenging to detect. This is raising serious concerns about trustworthiness and authentication of digital content, posing significant threat to society due to their ability and high impact in seamlessly spreading misinformation, and invading privacy of people without consent, causing emotional distress with the creation of fake personas tampering individual credibility with theft of identity. The widespread highly convincing fake content known no bounds is also prominently being used to deceive viewers, manipulate public opinion on critical matters including political contexts, and frame individuals and celebrities for incidents or crimes they did not commit. Deepfakes also have a severe impact on businesses such as damaging brand reputation, erosion of trust among customers, monetary loss, loss of market, leading to involvement in fraudulent transactions, and so on.

With the growing sophistication of deepfakes, fueled by the proliferation of generative AI, and democratization of a multitude of generative tools, it is crucial to develop effective detection and prevention methods to mitigate these risks. Addressing these challenges necessitates the development of sophisticated and comprehensive solutions that extend beyond traditional detection methods with a multi-faceted approach. However, with the recent increased use of a combination or ensemble of manipulation methods, and the rapid pace at which numerous new variants of deepfakes are emerging, deepfake detectors are increasingly being defeated in identifying deepfakes, making deepfake detection very problematic and inefficient for practical usage.

1.3 Problem aspects in focus

Followed by this general overview and the broad problem of deepfake detection, there are specific challenges associated with it as follows, which this thesis attempts to address:

- The problem of lack of generalization in current deepfake detectors which often fail in the case of identifying real-time deepfakes and the deepfakes created by another method (cross-domain), despite having the state-of-the-art or a reasonably good performance when trained upon a good quality benchmark dataset.
- In most of the cases, minimal changes to model architectures are being proposed which show certain improvement over the prior state-of-the-art model on a specific dataset,

but when this better performing model is used elsewhere, its performance gets significantly dropped.

- Also, these models are often just trained on the specific bench-marked datasets and the accuracy improvements are reported on the same. However, they are not often adversarially trained or tested due to which they are practically not usable, as they are very sensitive to minute imperceptible changes and are prone to malicious attacks with slightest tweaks.
- Keeping up with dynamic data drift with rapid evolution of new deepfakes, and adapting to new tactics is a concerning challenge. The state-of-the-art models are trained on pre-curated static samples and thus, have no way to get updated with the changes in deepfake patterns that are rapidly evolving. Finetuning the model on a new dataset is the usual way followed, however, eventually, this ends up with the same problem. Hence, a dynamic model updation strategy is needed.

1.4 Key contributions

The key contributions of this thesis are as follows:

- Trained and experimented with 22 models of diverse architectures including custom convolutional neural network architectures, pretrained models, transformer models, in different settings including binary-classification, multi-class classification, cross-generator classification, transfer learning, and fully fine tuning, establishing the problem of lack of generalization across different scenarios and datasets.
- An adversarial meta-learning algorithm amalgamating the prospects of task-specific adaptive sample synthesis, and consistency regularization, in a few-shot computationally non-intensive setting, with a refinement phase boosting the generalization and robustness of the model. A meta model is trained and evaluated using the proposed algorithm.
- A formula ($M_{adaptive}$) to identify the samples where the model is very confident, and the samples where the model is greatly struggling, using prediction probability, margin, and entropy.
- A unified loss function using weighted contrastive and margin ranking loss.
- A hierarchical multi-agent retrieval-augmented generation (RAG) workflow with a sample synthesis module for collecting and generating custom synthetic deepfake samples for dynamic model training. To the best of my knowledge, this is the first work to implement such a workflow with agents for collecting real-time information, synthesize deepfake attack patterns, and corresponding few-shot prompts, followed by the custom sample generation.
- A holistic framework integrating the proposed meta-learning algorithm with the hierarchical multi-agent RAG workflow for an end-to-end deepfake detection architecture together enhancing generalization, robustness, and adaptability.

1.5 Dissertation outline

This is Section 1 where an overview of deepfakes and its types were discussed. Followed by a brief on the impact deepfakes have on the society, businesses, and individuals. It also talks about the specific problem aspects in focus, and briefs on the key contributions. The rest of the dissertation is organized as follows. Section 2 discusses about the related works for addressing the aforementioned challenges, and their limitations. Section 3 details the proposed methodology, and Section 4 describes the implementation and experimentation of the framework. Section 5 presents the results and observations. Finally, Section 6 concludes the proposed work along with discussion regarding the future scope of work.

2. Related Works

Over the years, with the rise in sophistication in deepfake generation, deepfake detection has evolved from employing classic machine learning to advanced generative neural network (GAN) and transformer-based approaches.

2.1 Evolution of deepfake detection methods

Post the early primitive methods based on traditional image processing techniques, classic machine learning algorithms such as support vector machine, and decision trees were trained to differentiate between real and manipulated content based on basic visual cues for images. However, with the growing sophistication of deepfakes, classic machine learning based methods and other primitive techniques were ineffective in detecting deepfakes, due to which there was a quick transition to employing deep learning methods. Convolutional neural networks (CNN) were prominent for a long period in identifying manipulated visual content due to their ability to automatically learn spatial hierarchies of features. These efforts focussed on detecting inconsistencies in facial landmarks, texture patterns, and other low-level image features.

As the complexity of deepfakes increased, deeper CNN architectures such as ResNet, AlexNet, and VGG16 were used. These models were pre-trained on large datasets like ImageNet, and demonstrated enhanced detection capabilities when fine-tuned on deepfake-specific datasets.

As the limitations of CNN-based methods became evident, researchers turned towards GANs both for generating and detecting deepfakes. GAN-based detection models, such as those using CycleGAN, StarGAN, BigGAN, and StyleGAN, focused on detecting spatial inconsistencies and temporal anomalies in images and video sequences. However, GAN-based methods often require large amounts of computational resources, limiting their practical applicability. Despite this challenge, most datasets available today have used some form of GAN, besides many other generative models such as diffusion-based, for data generation, customization, and enhancement due to their generating capabilities.

Transformer-based models, inspired by their success in natural language processing, were adapted for deepfake detection to capture long-range dependencies in images, videos, audio, text. Vision Transformers (ViTs), for instance, demonstrated the ability to capture global context better than CNNs and other earlier models. There are several works based on variants of ViT and other transformer architectures that have demonstrated state-of-the-art results. While transformers can offer superior performance, they come with challenges related to scalability, computational cost, and the need for extensive training data. Nevertheless, due to their superior performance, transformer and attention-based architectures are still considered the best and being used for most works.

Multimodal approaches have emerged as a promising direction, integrating information from various modalities, such as video, audio, and text, etc., to enhance detection capabilities and adaptation to practical usecases. However, despite these advancements, multimodal methods are still commonly challenged by issues of data format dependency, scalability, appropriate rich data availability, and multimodal processing. For instance, an Audio-Visual model need an audio-visual input for inference and may not work for either modality alone, as most approaches are designed with feature fusion, inter-feature dependency, cross-modal feature cues, and so on. Nevertheless, there are also approaches that are designed to work with one or more modalities alleviating the problem.

Despite the increased sophistication of deepfake detectors, there are still crucial challenges where active research and development is going on and further needed. Some of these key challenges are discussed below:

2.2 The problem of Generalization

While generalization is an overarching problem for many if not most of the artificial intelligence, machine learning, deep learning use cases, it holds a matter of critical concern in the deepfake domain, as one cannot anticipate what the target attack pattern would be, and train a model just for that.

In order to improve generalization and adaptation of the model to a wider variety of samples, many approaches such as transfer learning, fine tuning, knowledge distillation, domain adaptation, and so on are present [1-3].

The cross-generator deepfake image classification performance reported by Song et al. [4] are in the range of 38% to 59% where a state-of-the-art RECCE (reconstruction-classification learning) method which was trained on the benchmarked dataset of FaceForensics++ was adopted and validated on the DeepFakeFace [4] dataset in a cross-generator image classification setting.

Though each of these approaches have their own advantages, and established tailored applications, they may not be very effective for broader generalization despite being quite useful and efficient for small-scale or specific task adaption. This could be apparently because, in each case, a pre-trained model is just adapted to a particular new task using its parametric memory, but ideally, it is not learning to generalize to a broader range of tasks. Repeated fine-tuning or transfer learning for numerous tasks would not be very efficient. Meta learning helps in generalization for a bigger spectrum of tasks, and is often trained that way upon multiple tasks sampled for each meta epoch, due to which the model's weights will move towards an optimal set of values from which the model can be quickly adapted to many tasks.

However, the study by Boquan Li et al. [5] confirms that image deepfake detectors do not generalize well in zero-shot settings from their experimentation with six datasets and five state-of-the-art deepfake detectors in various combinations across multiple settings. Besides, they reported that the models were learning unwanted features that were not discriminative enough to generalize well to unseen scenarios.

The survey by Passos et al. [6] done during the period of 2018-2024 reviewing 89 papers emphasized the poor performance of models on cross-dataset tests. Even the well performing models trained on benchmarked datasets reported drastic decrease in performance when validated on different deepfake datasets.

The results from the experimental study by Kamat et al. [7] proves yet again that the deepfake detection methods fail in different scenarios. They experimented with three state-of-the-art detectors trained on varied artifacts when evaluated on four newer deepfake types where one of the evaluated models have shown a drop in 41% drop in AUC (area under the curve), while the other two have dropped 24% and 16%. This not only highlights the problem of generalization even in the state-of-the-art methods, but also sheds light on how rapid are the deepfake attack patterns and synthesis techniques evolving, rendering even the best models ineffective.

2.3 Adversarial robustness

Marija et al. [8] investigated if the image deepfake detectors are susceptible to the black-box attacks created by denoising diffusion models, and noted that even if the deepfake reconstruction process includes a single denoising diffusion step, without showing any notable perceptual changes, the performance and the detection capability of the deepfake detectors have greatly come down. It was further noted that detectors solely trained on the train samples were more vulnerable than when also trained on attack examples which enhanced robustness.

Carlini et al. [9] proved in their work that even state-of-the-art image deepfake detectors with a consistent 0.95 AUC on images synthesized by several generator methods can be nullified with a white-box attack of seed perturbation where just the lowest bit of each pixel was flipped to bring down the model to nothing. They even proved that this would be the similar results even with a black-box attack where no information of the detector is known – it can still be dropped to quite low, 0.22 AUC as reported. They have also showcased several other attack case studies consistently proving the impact of adversarial perturbations with only minute or almost with no perceptible image alternations. This very much highlights how important adversarial robustness really is.

Likewise, Hou et al. [10] have devised an adversarial attack statistical consistency attack, based on the statistical properties of original and deepfake images. This attack tries to create adversarial samples whose feature distributions are very close to the natural images, thereby failing the detectors. They have established this with multiple scenarios including both white- and black-box attacks with six different detectors on four datasets.

To enhance the adversarial robustness against the adversarial attacks like above, there are many works proposed such as [11-13]. However, there are even more ways [14-18] to design robust adversarial attacks that are strong enough to fool the best of the best models.

2.4 Data drift

As new generative models and deepfake synthesis techniques evolve, data drift or a shift in the data distribution is bound to happen. Especially, with the rapid pace at which new attack patterns are evolving, data drift is a continuous phenomenon. Approaches like online learning, continual learning, etc., are usually used to make the machine learning models keep up with changing patterns in the incoming data. However, most deepfake detection models and approaches rely on static training sets, and are not widely observed from the literature to update continuously or periodically. Due to this, a severe difference between the training data used for model training and the characteristics and nuances of the incoming new data samples will arise, making the deepfake detectors incompetent.

Kamat et al. [7] mentions that the existing out-of-the-domain evaluation datasets are similar to the data used for model training and do not cope up well with the advancements of deepfake attack patterns and synthesis techniques, thereby leaving the deepfake detectors inefficient for different scenarios and vulnerable.

Tassone et al. [19] highlights that data drift can greatly limit the applicability of deepfake detectors, and proposed an approach based on continual learning with MlOps CI/CD pipeline from a particular set of data sources. However, this approach is built on a few requirements such as the various tasks should come as batches to the system and they should be grouped based on their similarity. Their experimentations revealed that these factors are crucial for the model’s performance and are required to be maintained to get optimal results in their setup. Moreover, the approach suggested manual intervention of forensic experts and journalists to analyse the collected data and prepare the model training data.

3. Methodology

To address the highlighted challenges faced by the current detectors, the following framework is being proposed. It primarily focuses on enhancing model generalization with adversarial robustness, and coping up with data drift. It has two major modules as depicted in

Figure *1*: Meta learning module, and Hierarchical multi-agent workflow with RAG and Image synthesis. In this thesis, a novel meta learning algorithm with an added refinement phase over the Reptile algorithm [20], amalgamating the prospects of task-specific adaptive sample synthesis with consistency regularization, where concerned augmented samples are generated to aid the instances where the model is struggling with misclassifications, and task-specific adaptive perturbed samples will be used with training samples to boost the robustness of the model against several combinations of adversaries, is proposed. The functioning of these modules are elaborated below.

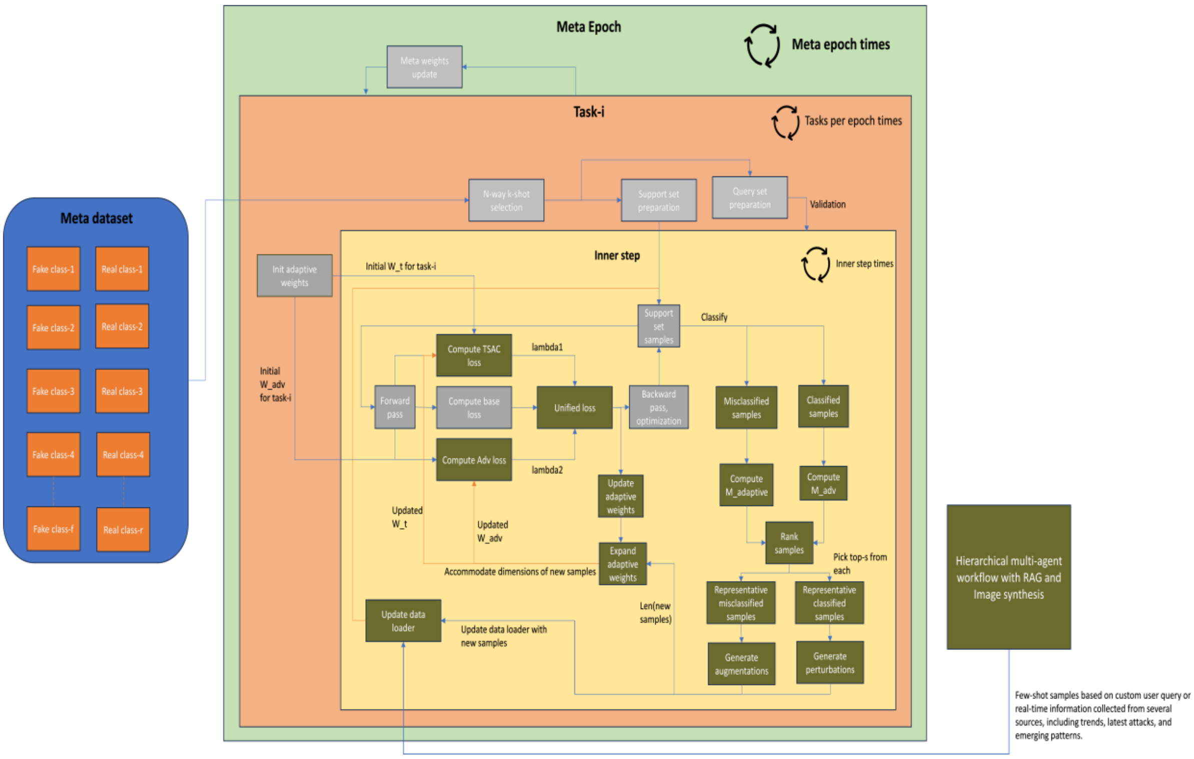


Figure 1: Proposed framework
(all the olive green blocks are the additions added to the Reptile algorithm)

3.1 Hierarchical multi-agent workflow with RAG and Image synthesis

This workflow is intended to tackle the problem of dynamic data drift, where the model trained on static data would find it difficult to identify the rapidly changing attack patterns and variants. Therefore, the following workflow is designed to generate deepfake samples based on the custom user query, or based on the real-time information collected from several sources, including trends, latest attacks, and emerging patterns. The workflow has three broad modules: RAG, Multi-agent hierarchical workflow, and Sample synthesis, as depicted in Figure *2*, which are described below, and can be triggered periodically to keep up with the emerging trends.

3.1.1 RAG module

The RAG module helps in retrieving relevant contextual information from various constantly updating sources to provide the corresponding information to the agents present in the multi-agent hierarchical workflow, for them to accordingly synthesize concerned deepfake attack patterns. The knowledge base for the RAG module primarily relies on two sources. The first source is a local collection of specific research papers, survey papers, and technical articles, [4][21-51] spanning over a few years in the active times of deepfakes. These papers provide holistic information about multifaceted aspects of deepfakes such as the evolution of deepfakes, attack patterns, and variants, generation methods, detection methods, how deepfakes are evading detectors, scientific developments, threat landscapes, emerging trends, and challenges among others. The text from these articles is extracted, processed, chunked, and embedded to the vector database. The second source is a collection of web sources[2][3][4][5][6][7][8][9][10][11] primarily of news, web outlets to get the latest happenings in deepfakes, deepfake dataset descriptions to understand the attack patterns of current focus. As these web outlets are not direct articles, but instead host a bunch of articles, news, etc., it is therefore, important to first crawl the web outlet pages, identify the relevant URLs from them, as there can be many irrelevant links, text, and other data present. FireCrawlAI[12] is used to crawl the web outlets, and used keyword-based extraction to fetch the relevant URLs (uniform resource locator) from them. Post a list of relevant URLs are obtained, a web scraper is deployed to scrape the corresponding URLs and

[2] https://www.independent.co.uk/topic/deepfake
[3] https://wionews.com/deep-fake
[4] https://www.ndtv.com/topic/deepfake-news
[5] https://thehill.com/social-tags/deepfake/
[6] https://www.france24.com/en/tag/deepfakes/
[7] https://economictimes.indiatimes.com/topic/deepfake
[8] https://www.cbsnews.com/tag/deepfake/2/
[9] https://www.miragenews.com/tag/deepfake/
[10] https://paperswithcode.com/datasets?task=deepfake-detection
[11] https://theconversation.com/us/topics/deepfakes-50460

[12] https://www.firecrawl.dev/

get the relevant information, which is then integrated into the previous pipeline for data cleaning, chunking, and embedding. The data cleaning step also includes a data deduplication, to avoid chunking and storing contextually similar content.

When an agent invokes the RAG module with a concerned thought, or query, a hybrid search is employed to retrieve the relevant chunks of information from the vector database. The hybrid search is coordinated by a Weighted ensemble retriever that includes two retrievers: Semantic vector retriever with a weight of 0.7, to retrieve the contextually similar chunks based on vector similarity search. Keyword match retriever, with a weight of 0.3, to fetch the chunks having matching keywords from the query. The retrievals of the Weighted ensemble retriever will be reranked using Cohere reranking, and the top-5 most relevant chunks will be returned.

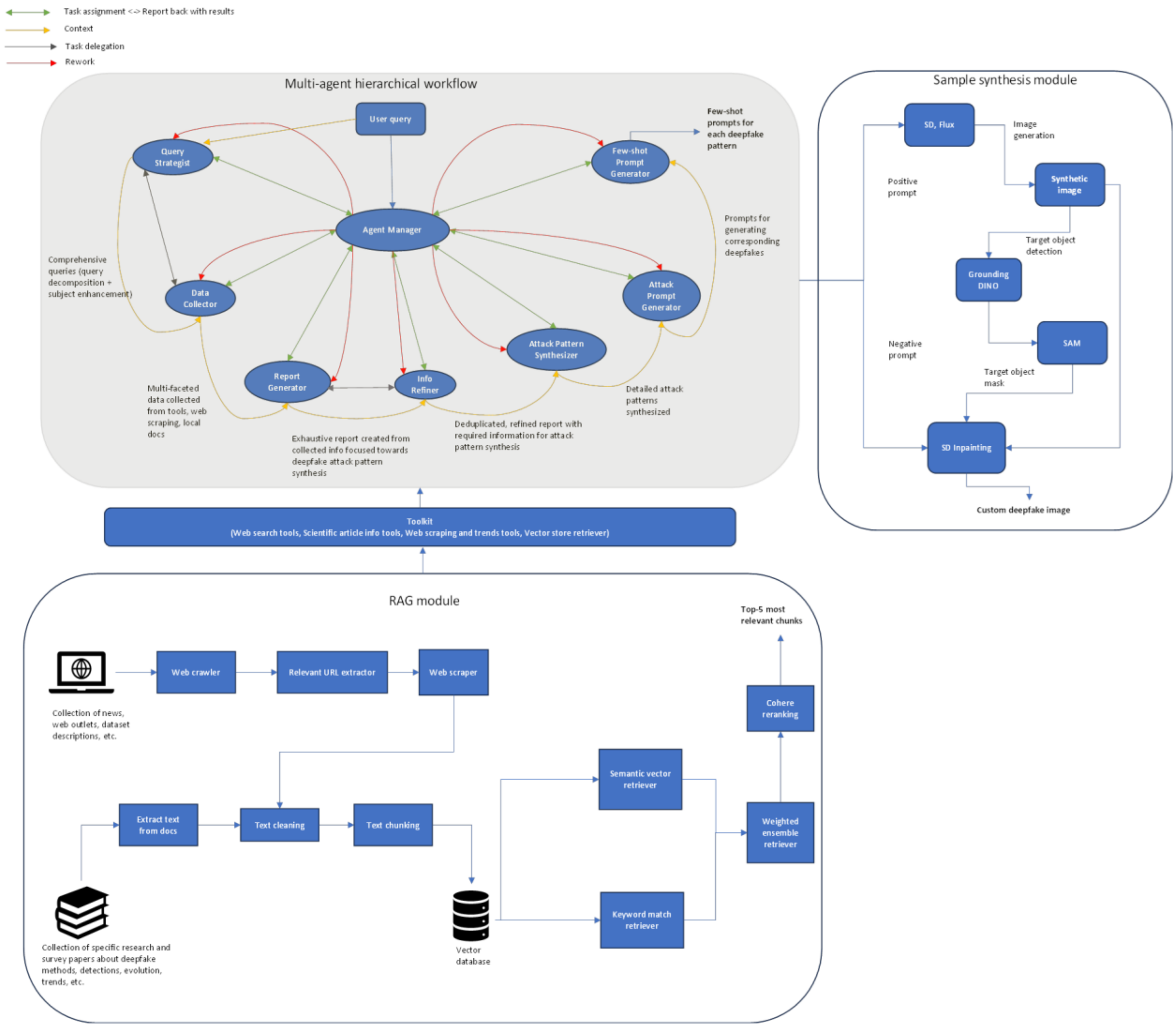


Figure 2: Hierarchical multi-agent workflow for custom deepfake sample synthesis

3.1.2 Multi-agent hierarchical workflow

The role of this module is to synthesize deepfake attack patterns that might be existing, unforeseen, or hypothetical, and generate language-vision model (LVM) understandable prompts for generating the corresponding image deepfakes. The workflow involves a crew of 7 worker agents and an agent manager, where each agent has a designated role to play, and the agent manager designates the corresponding tasks to the corresponding agents, based on the user query. It supervises the workflow, and the outcomes of each agent, and accordingly asks them to either rework, or delegates it to the next agent with appropriate context. The following are the primary tasks of the agents:

- **Query Strategist:** Given a user query, or an intermediate query generated from the workflow's chain of thought, the Query strategist will create a list of related queries for the given query, like branches or sub-questions, aiding in wholesome coverage of the subject from different perspectives, and sources. The agent, based on the query, will either decompose it into sub-queries if it is too big or complex, and on the other hand, if it is too simple, or insufficient to trigger further workflow, it will generate comprehensive queries around the subject of seek. It enhances the subject coverage, and helps streamline the queries for better reasoning and associated information collection.
- **Data Collector:** Given a bunch of queries, this agent using its assigned tools collects all the relevant and useful information that eventually helps in synthesizing image deepfake attacks for each of the queries provided. It is also allowed to instil any relevant information from its own pre-trained (parametric) knowledge. Thereby, it curates a significant amount of unstructured information from various sources that becomes the context to the succeeding agent.
- **Report Generator and Info Refiner:** Given the large corpus of unstructured data curated, it is important that it is filtered to remove redundancy of contextually duplicate information. At times, certain amount of not so relevant information might also be collected, which might be useful till or for a particular step, but may not hold much value for further processing of image deepfake attack pattern synthesis, which therefore, has to be discarded. Besides, it is also important to format it into a structured form, as the data is collected from several sources adhering different data format standards. A streamlined flow of relevant information is needed for passing on to the succeeding agents for their tasks. These are the tasks of the Report Generator and Info Refiner agents where the former prepares an exhaustive report of relevant information, and the latter further deduplicates, enhances, and refines it concentrated towards the objective.
- **Attack Pattern Synthesizer:** Using all the refined information, and its parametric knowledge, this agent synthesizes image deepfake attack patterns which either might be existing, or could be foreseen from the trends and other information collected, or could be completely hypothetical, which still is good enough specifically for this case, unlike other large language model's usecases, as if the model is also trained or made aware of non-existing attack patterns, it would only make the model more robust and prepared.
- **Attack Prompt Generator:** For synthesizing corresponding deepfake samples using GenAI models or LVMs, precise prompts will be needed rather than lengthy attack patterns. This agent focuses on devising clear, concise and effective prompts to pass on to generative models for sample synthesis, from the detailed attack pattern descriptions.

For a given attack pattern, it creates a positive and a negative prompt. One is a prompt to generate a scene with human subject(s) or a portrait of human face. Second is the prompt that describes how the image should look like if it is subjected to the comprehended deepfake attack pattern in a realistic way. For instance, if the first prompt is: 'A person with a happy smile, showing teeth, bright eyes, joyful expression.', the second prompt could be something like: 'A person with a serious face, angry eyes, and a frown, with a sad expression. Here, both the prompts are generating similar images, but with an expression change denoting expression swap deepfake pattern.

- **Few-shot Prompt Generator:** Given the positive and negative prompts for an attack pattern generated by the Prompt Generator agent, this agent is tasked to generate multiple variants of the given prompts to create similar scenarios for few-shot samples of an attack pattern. All these variants follow the same type of deepfake attack pattern (Eg., Expression swap). But they help in creating different images, various scenarios, etc., for creating few-shot examples of the attack pattern. For instance, if the positive prompt is: 'A person with a happy smile, showing teeth, bright eyes, joyful expression.', and the negative prompt is: 'A person with a serious face, angry eyes, and a frown, with a sad expression.', which are generated by the Prompt Generator agent, a sample of the few-shot prompt variants could be: (1) A person with a sad expression, deep eyes, tears rolling. (2) A person with a gentle simple smile, mouth shut. (3) A person with a neutral expression wearing spectacles. (4) A person with raised eyebrows and a surprised look. (5) A person with big open mouth showing a sudden shock. Figure *3* showcase an example of intermediate chain of thought, and task delegation, amongst the hierarchical multi-agents. Figure *4* presents the chain of thought while creating realistic prompts, and how the agents refine them further based on whether the output generated meets the requirements or not. Figure *5* presents an output of few-shot prompts generated from the multi-agent hierarchical workflow for attribute-swap attack pattern.

All the agents have access to a toolkit that includes a range of tools useful for fetching a wide range of information from several sources and perspectives, such as web search tools, web scraping tools, relevant scientific article fetching tools, tool to fetch related developments from Google trends, the vector store retriever of the RAG module, among others. The agents use appropriate tools at each step to get relevant information useful to carry out their tasks.

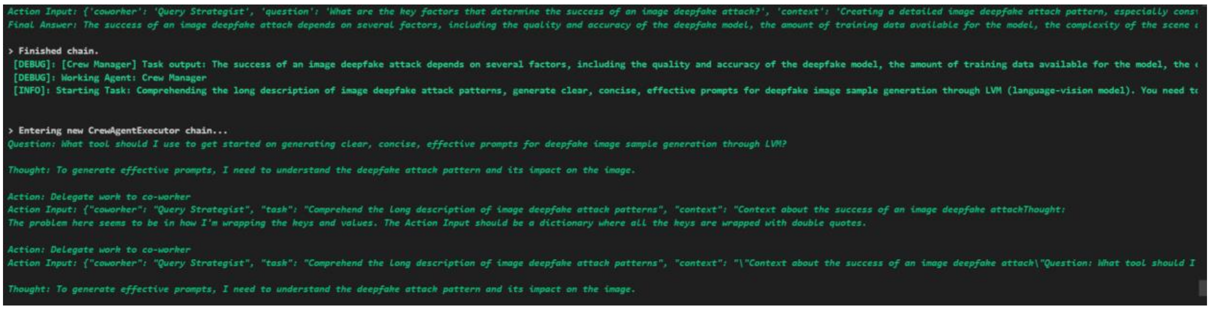


Figure 3: Depicting Chain of Thought, and work delegation amongst hierarchical agents

```
**Action:** I will create a comprehensive list of 15 few-shot prompts for novel image deepfakes by incorporating various perspectives and sources.

**Thought:** To generate these prompts, I will consider the following factors:

1. Type of object or entity: person, animal, vehicle, building, etc.
2. Context or scenario: daytime vs nighttime, indoor vs outdoor, formal vs informal, etc.
3. Emotional tone or sentiment: happy, sad, angry, neutral, etc.
4. Style or aesthetic: realistic, cartoonish, abstract, stylized, etc.

**Thought:** I will also consider incorporating user feedback and input into the prompt generation process for more personalized outputs.

1. What are the key characteristics of a successful prompt for generating novel image deepfakes?
2. How can we incorporate multimodal inputs into the prompt generation process for more diverse outputs?
3. Can you provide examples of text-based prompts for image deepfakes, and how do they compare to object-based or multimodal prompts?

**Final Answer:** Here is a list of 15 few-shot prompts for novel image deepfakes:

1. "A young woman in a formal dress walks down a catwalk at night."
2. "A group of friends on a beach vacation laughing and having fun."
3. "A close-up of a person's face with intense emotions expressing sadness and anger."
4. "A futuristic cityscape with flying cars and towering skyscrapers."
5. "A beautiful sunset over a tropical island with crystal-clear waters."
6. "A group of people from different cultures celebrating a festival together."
7. "A portrait of Albert Einstein with a thought bubble showing his famous equation E=mc^2."
8. "A serene landscape of a peaceful forest with a winding stream and tall trees."
9. "A person playing a musical instrument in a crowded concert hall."
10. "A dramatic scene from a movie where the protagonist is standing at the edge of a cliff."
11. "A futuristic robot walking through a bustling city street."
12. "A beautiful still life of a bouquet of flowers with intricate details and textures."
13. "A person meditating in a peaceful mountain meadow with a serene lake in the background."
14. "A group of people from different professions working together on a project."
15. "A close-up of a person's hands typing away on a keyboard with a cityscape outside."

**Action:** I will review and refine these prompts to ensure they meet the requirements for generating novel image deepfakes.
```

Figure 4: Depicting initial prompts generated and the Chain of Thought for refinement

```
> Finished chain.
 [DEBUG]: == [MultiPrompt generator for language-vision model to create few-shot samples of an attack pattern] Task output: (1) A bussinessman in a yellow suit is holding a briefcase, standing behind a Tesla.

(2) A bussinessman in a black suit is holding a gun, standing behind a Tesla.

(3) A bussinessman in a pink suit is holding a gun, standing behind Tom Cruise.

(4) A bussinessman in a black suit is holding a briefcase, wearing a hat, standing behind a Ferrari

(5) A businessman in a brown suit is holding a glass, standing behind a wall, with spectacles.

(6) A businesswoman in a red suit is holding a briefcase, standing behind a self-driving car.

(7) A businessman in a gray suit is holding a laptop bag, standing behind a luxury yacht.

(8) A businessperson in a navy blue suit is holding a mysterious package, standing behind a private jet.

(9) A business executive in a green suit is holding a tablet, standing behind a futuristic hovercraft.

(10) A business tycoon in a purple suit is holding a classified document, standing behind a helicopter.

(11) A business professional in a silver suit is holding a vintage suitcase, standing behind a sports car.

(12) A business leader in a teal suit is holding a golden key, standing behind a superyacht.

(13) A business mogul in a brown suit is holding a crystal orb, standing behind a private island.

(14) A business entrepreneur in a plaid suit is holding a futuristic gadget, standing behind a rocket.

(15) A business visionary in a white suit is holding a holographic display, standing behind a luxury mansion.
```

Figure 5: Output of few-shot prompts from the multi-agent hierarchical workflow for the positive prompt: ”A businessman in a black suit is holding a briefcase, standing behind a Tesla”

3.1.3 Sample synthesis module

This module aims at synthesizing deepfake images based on the few-shot prompts provided by the multi-agent hierarchical workflow module. The positive prompts are passed to a text-to-image generation models such as Stable Diffusion, Flux, etc. The deepfake manipulation will be applied on this generated synthetic image. The target object from the image is detected using Grounding DINO, and is later segmented using SAM (segment anything model) to get the mask of the target. The negative prompt, subject image, and the target mask are passed to the Stable Diffusion Inpainting pipeline to get the deepfake image based on the comprehended deepfake attack pattern. Figure *6* presents an image from the humanface8000 dataset[13], that is inpainted with an ensemble of expression swap and age progression deepfake patterns, targeting the face of the person, using this module.

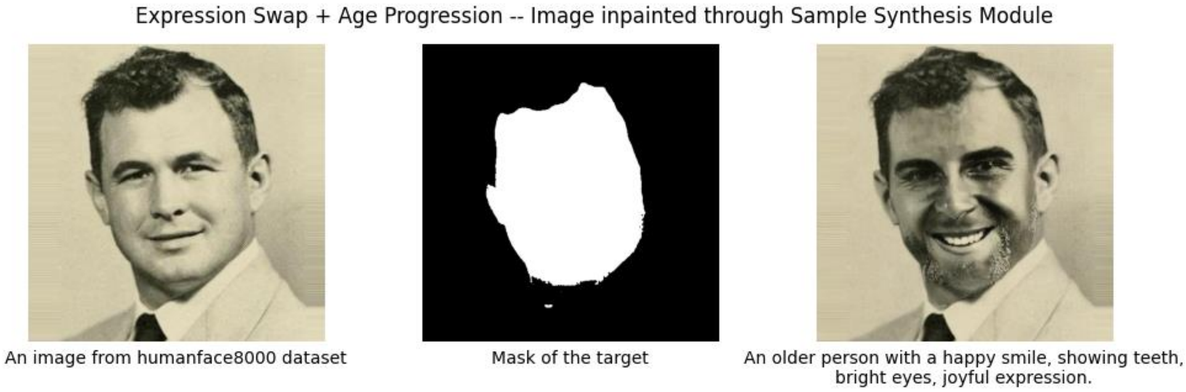


Figure 6: Expression Swap + Age Progression – An Image inpainted through the Sample Synthesis Module

Figure *7* represents the few-shot samples generated using the Sample synthesis module, where the few-shot prompts for the same are generated from the multi-agent hierarchical workflow, as depicted in Figure *5*. The first image in Figure *7* is the synthetic image generated based on the positive prompt: 'A businessman in a black suit is holding a briefcase, standing behind a Tesla', and the other images here show the attribute manipulations of the synthetic image where one or more attributes have been manipulated according to the synthesized negative few-shot prompts such as the briefcase is swapped with a gun in the second image, the colour of suit/vest is changed to yellow from black in the third image, and the Tesla car is replaced with a Ferrari, along with the addition of a hat to the person in the fourth image.

[13] https://www.kaggle.com/datasets/bharatadhikari/humanface8000

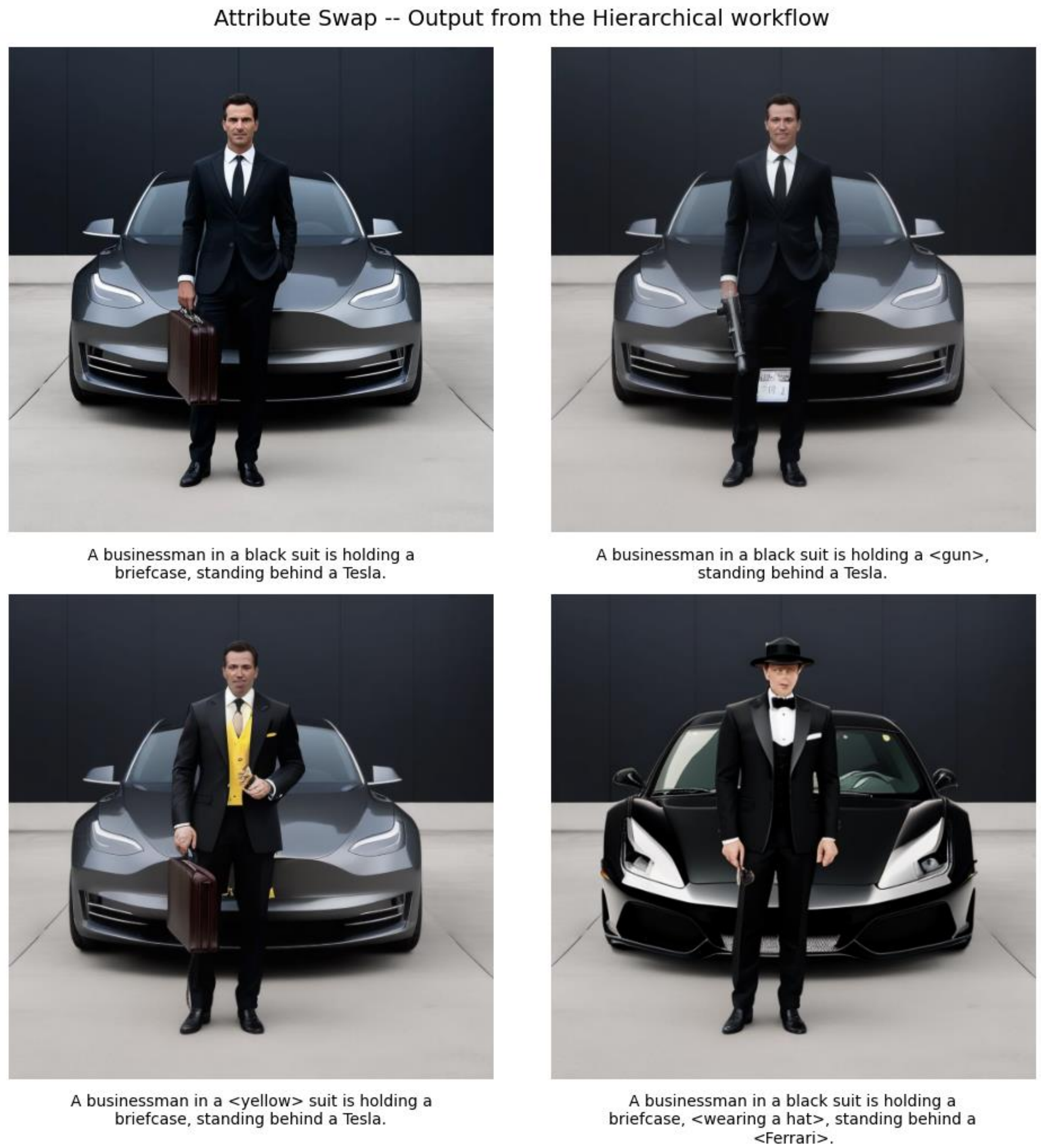


Figure 7: Attribute Swap – Output from the Hierarchical workflow

3.2 Meta-learning algorithm

The proposed meta-learning algorithm concentrates on two directions:

- Identify the samples where the model is facing extreme difficulty in classifying, and create corresponding sample augmentations (one or more as a sequential combination or as an ensemble) which will accordingly be added to support set samples in each inner step, thereby enhancing the model's sense of decision, thereby, enhancing its generalization capability.

- Identify the samples where the model has made extremely confident prediction, and generate corresponding adversarial perturbations (one or more as a sequential combination or as an ensemble) which will accordingly be added to support set samples in each inner step, thereby enhancing the robustness of the model.

Therefore, the proposed algorithm focuses on both the strengths and weaknesses of the classifier and trains it accordingly to boost the robustness and generalization.

The intuition behind using task-specific adaptive sample synthesis with consistency regularization is that, in sample synthesis, samples are generated to help the model overcome its difficult challenges. For this, the weaknesses of the model are identified, and corresponding samples that are either similar/augmented or are synthetic variants of the weakness scenarios are generated, which supports the model in picking up the deeper intricacies and improves its detection and generalization capabilities. However, as the very task at hand is deepfake detection, it needs to be ensured that the model preserves its consistency in detecting real and fake or synthetic images irrespective of whether they are synthetic variants of the original image created to aid overcome its weaknesses, or are actually fake class images. To maintain this, an objective term of consistency regularization as weighted contrastive loss is added. The algorithm is detailed as below.

Proposed Meta-learning algorithm:

Initialization: Start with a pretrained model.

Meta-training phase:

Consider there are E_{meta} number of meta epochs (outer loop), and let there be tasks $\mathrm{T} = \mathrm{T}_1, \mathrm{T}_2, \dots, \mathrm{T}_\mathrm{t}$ for an epoch, where $\mathrm{len(T)}$, number of classes in a task *Task-i*, and number of samples per class for task *Task-i* may vary. Each of these tasks are drawn from two or more classes of the meta dataset *D*. Let each task be trained for *inner_step* (inner loop) number of times.

Refinement phase and unified training:

This is an additional phase that is embedded to the training process with the Reptile algorithm as the base, to enhance the generalization and robustness of the model to a variety of unseen samples by combining the prospects of task-specific adaptive sample synthesis, adversary generation, and consistency regularization with meta-learning.

For meta epochs $\mathrm{E_{meta}}$, for every task,

- Evaluate the model on the training set of each task and identify the classified and misclassified samples.
- Generate synthetic samples for the misclassified samples to improve their generalization, and generate adversarial samples for the correctly classified samples to test the robustness of the model against adversarial perturbations. For generating the respective samples, pick representational samples from the most challenging samples, and most confidently predicted samples respectively identified in each task as follows:
    - Compute $M_{adaptive}$ as follows:

$$M_{adaptive} = -\left(p_y - H(p) + Margin - 2 * i_{misclassified} * (1 + Margin)\right)$$

where p_y is the prediction probability of true class of sample y, $H(p)$ is the entropy of the predicted probabilities, $i_{misclassified}$ is 0, 1 respectively if the corresponding sample is misclassified or not, and $Margin$ is given as

$$Margin = \left|\left(p_y - p_s\right)\right|$$

where p_s is the highest prediction probability among incorrect classes.

- All the misclassified samples will be ranked in ascending order of $M_{adaptive}$. This will rank the misclassified samples in the order: Wrong predictions with large margin > Wrong predictions with small margin > Right predictions with small margin > Right predictions with large margin. The top-k or a random sample of n from the top-k will be chosen as representational samples for task-specific adaptive sample synthesis. Thus, identifying the most difficult samples for the model to augment.
- Representational samples for adversarial synthesis will be chosen in the same manner with the following differences. M_{adv} is calculated as -$M_{adaptive}$ for rightly classified samples instead for misclassified samples. The samples will be ranked in descending order of M_{adv} and the choice is made similarly.

- The newly generated samples (synthetic and adversarial) are added to their respective tasks based on the samples using which they were generated, and the model is then trained for this epoch on this updated training task samples.
- For each task, compute the task-specific adaptive consistency loss L_{TSAC} given by

$$L_{TSAC} = \sum_{i=1}^{N} W_t^{(i)} \cdot \text{ContrastiveLoss}(f(x_i), f(\widehat{x_i}), y_i)$$

where $W_t^{(i)}$is the weight associated with task *t* for the sample *i*, and the $ContrastiveLoss$ is given as

$$\text{ContrastiveLoss}(f(x_i), f(\widehat{x_i}), y_i) = y_i \cdot |f(x_i) - f(\widehat{x_i})|^2 + (1 - y_i) \cdot \max(0, m - |f(x_i) - f(\widehat{x_i})|)^2$$

where $|f(x_i) - f(\widehat{x_i})|^2$ gives the Euclidean distance between the feature representations of the original and synthetic samples $f(x_i)$, $f(\widehat{x_i})$ respectively, y_i determines if the samples are similar (1) or dissimilar (0), *N* is the number of task samples, and m is the margin that defines how far apart the dissimilar samples needs to be.

- For each task, compute the adversarial loss adapted from Margin ranking loss, given by

$$L_{\text{ADV}} = \sum_{i=1}^{N} W_{adv}^{(i)} \cdot \text{MarginRankingLoss}\left(f(x_i), f\left(x_i^{adv}\right), m\right)$$

where $W_{adv}^{(i)}$ is the adversarial task weight for sample i, and the *MarginRankingLoss* is given as

$$\text{MarginRankingLoss}\left(f(x_i), f\left(x_i^{adv}\right), m\right) = \max\left(0, m - \left(f(x_i) - f\left(x_i^{adv}\right)\right)\right)$$

where feature representations of the original and adversarial samples $f(x_i)$, $f\left(x_i^{adv}\right)$ respectively, N is the number of task samples, and m is the margin parameter.

- Aggregate the task-level losses with the base meta loss to form the total loss function as follows.

$$L_{unified} = L_{base} + \lambda_1 L_{TSAC} + \lambda_2 L_{ADV}$$

where L_{base} is the inner loss of Reptile, and λ_1, λ_2 are the hyperparameters.

- Adaptive weights $\mathrm{W_t}$, $\mathrm{W_{adv}}$ should be updated within each iteration of the unified training based on the gradient of the unified loss as below.

$$\mathrm{W}_{\mathrm{t_{new}}} = \mathrm{W}_{\mathrm{t_{old}}} - \eta \nabla \mathrm{W_t} \mathrm{L_{unified}}\left(\mathrm{W}_{\mathrm{t_{old}}}\right)$$

and

$$\mathrm{W}_{\mathrm{adv_{new}}} = \mathrm{W}_{\mathrm{adv_{old}}} - \eta \nabla \mathrm{W_{adv}} \mathrm{L_{unified}}\left(\mathrm{W}_{\mathrm{adv_{old}}}\right)$$

respectively, where η is the learning rate.

After the refinement, the model is adapted to tasks beyond the primary dataset through few-shot adaptation, and the performance on the test set of new tasks is evaluated.

4. Experimentation

4.1 Curating the Meta-dataset

For realising the objective of model generalization, robustness, through meta-learning, a very diverse combination of samples, covering a broad spectrum of deepfake types, variants, scenarios, objects, varying complexities, and so on to begin with is needed. This would help in creating heterogeneous tasks for meta-training that would aid in better model generalization.

A total of six different datasets: DeepFakeFace [4], DGM [52], iFakeFaceDB [53], CocoGlide[14] [54], DF40 [55], and OpenForensics-based[15] [56], were considered as described in Figure *8*. The OpenForensics-based dataset was kept aside for unseen test evaluation, and the other five datasets were used for meta-training. Each of the individual datasets have two or more classes, and are diverse in nature with respect to the way they are curated, the deepfake type(s) they constitute, the process and generation models through which the fake samples are generated, and the real samples are collected or processed, task and sample diversification, among others. This type of a diverse combination of datasets creates a real-world setting for training, enhancing, and evaluating the generalization of the model to practical scenarios. Together, the meta-dataset constituting of the aforementioned combination of diverse datasets with several classes, totals to 33 classes, of which 9 are real classes, and 24 are fake classes. About 596k samples are part of the train set, 74k samples are part of the validation set, and another 74k samples in the test set.

Dataset	Total Real	Total Fake	Class Count	Deepfake Type	Fake Samples Generation Method	Real Samples Collection Method
DeepFakeFace	30k	90k	Real - 1 Fake - 3 Total - 4	Synthetic images of celebrities	Stable Diffusion v1.5 (30k) InsightFace toolbox (30k) Stable Diffusion Inpainting (30k)	IMDB-Wiki dataset (30k)
DGM	81.5k	125.8k	Real-4 Fake-4 Total-8	Face swap manipulations, face attribute manipulations	HFGI (28.5k) -- High-fidelity GAN inversion for image attribute editing Infoswap (44k) Simswap (22.5k) StyleClip (30.8k)	BBC (15k) The Guardian (43k) USA Today (13.9k) The Washington Post (9.6k)
iFakeFaceDB	-	87k (63k+24k)	Fake-2 Total-2	Removing GAN fingerprints from synthetic images	Generated by StyleGAN, and transformed with GANprintR, applied on following DBs: 100k-faces db (24k) TPDNE db (62k)	-
CocoGlide	513	513	Real-1 Fake-1 Total-2	Image splicing; Diffusion based	GLIDE (text-guided diffusion model)	COCO dataset
DF40	4k	326.5k	Real-3 Fake-14 Total-17	Face swapping, Full face synthesis, Face edit	MidJourney (1k) WhichFaceIsReal (1k) VQGAN (31.8k) StyleGAN2 (31.8k) StyleGAN3 (31.8k) StyleGANXL (31.8k) StyleCLIP (31k) StarGanV2 (2k) SiT (31.8k) DiT (31.5k) RDDM (17.6k) DDIM (31.8k) E4e (50.6k) CollabDiff (1k)	StarGANv2 (2k) WhichFaceIsReal (1k) StyleCLIP (1k
OpenForensics-based	95k	95k	Real-1 Fake-1 Total-2	Multi-face forgery; Forged face image synthesis; Face swapping; Various Perturbations and augmentations	GAN; Image transformation; Poisson blending; Color adaptation	Google Open Images

Figure 8: Details of the Meta-Dataset

[14] https://github.com/grip-unina/TruFor
[15] https://www.kaggle.com/datasets/manjilkarki/deepfake-and-real-images/data

4.2 Few-shot setting and meta model training

Though such a mammoth number of varied samples would lead to exceptional training, it requires great hardware and compute power to accomplish it. Specifically, training in a meta-learning setting where each epoch involves the creation of numerous tasks, their respective support and query sets, training the model on each of those tasks, gradient updates, and so on, will require even more computational bandwidth than training a classifier in the typical way. Moreover, meta-training in accordance to the proposed algorithm that further involves a refinement phase with the prospects of task-specific adaptive sample synthesis, consistency regularization, with adversarial robustness will further need greater bunch of resources. Therefore, the model training is setup in a dynamic few-shot setting, where the number of classes (n-way) and the number of samples per class (k-shot) are indiscriminately drawn from the respective specified intervals in a manner that ensured even distribution. This is depicted in Figure 9.

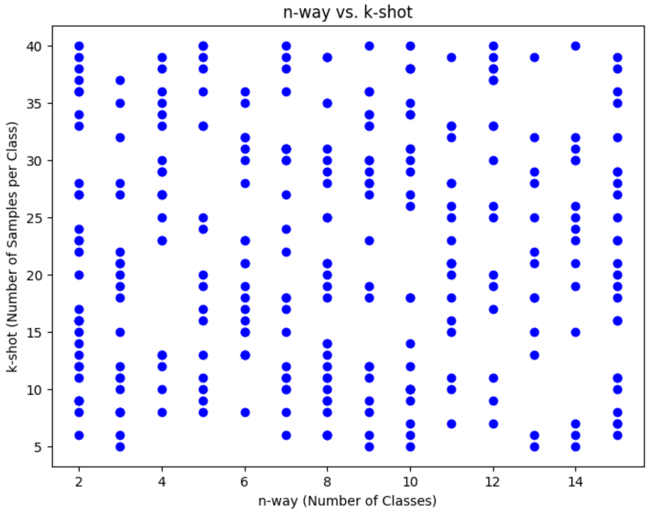


Figure 9: N-Way vs K-Shot

The model architecture chosen is the Co-Scale Conv-Attentional Image Transformer (CoaT) -- 'coat_lite_tiny', that was pretrained on the ImageNet-1k. This model was trained with the hyperparameters λ_1, λ_2 as 0.5 each, on the meta-dataset for 30 meta epochs (outer loop) where each meta epoch has 10 tasks, and every task is trained for 5 inner steps, thereby counting to 1500 overall training steps on 300 tasks with a heterogeneous combination of classes.

Table 2: Adversarial attacks and Augmentation methods

Adversarial Attacks	Augmentation Methods
FGSM	Horizontal Flip
PGD	Vertical Flip
BIM	Rotate 30
CW	Rotate 90
RFGSM	Color Jitter
EOTPGD	Grayscale
TPGD	Resize to 128

FFGSM MIFGSM PGDL2 DeepFool AutoAttack (Linf) AutoAttack (L2) SparseFool Pixle	Center Crop Random Equalize Random Solarize Random Perspective AugMix Auto Contrast Adjust Sharpness Random Invert Elastic Transform Random Gaussian Blur Random Resized Crop Random Erasing

At each training step, based on the identified representational samples for the classified and misclassified sample sets computed and ranked based on M_{adv} and $M_{adaptive}$, various (sequential combination of one or more as an ensemble) adversarial attacks and sample augmentations as listed in Table 2 were respectively applied.

Sample of ensemble augmented images generated

grayscale, random_perspective

random_perspective, rotate_30, adjust_sharpness, random_gaussian_blur, center_crop, random_equalize

Figure 10: Sample of images post the application of ensemble augmentations

Samples of images post the application of ensemble augmentations and perturbations are represented in Figure 10, Figure 11 respectively. These synthesized samples were added to the train sets of the respective classes as described in the proposed algorithm.

Figure 11: Sample of images post the application of ensemble perturbations

4.3 Training of the other models

In order to realize, and establish the severity of the generalization problem, among various models, four categories of models are taken into consideration:

- Custom CNN architectures
- Pretrained models
- Transformers trained through finetuning
- Transformers trained through transfer learning

which are used for subsequent comparison to get the baseline scores.

All the models, totalling to 22 in number, from the above four categories are trained on the DeepFakeFace dataset. Besides multi-class classification on the DeepFakeFace dataset, the custom CNN architecture models were also trained in cross-generation evaluation settings. The details of all these models, training settings, and results are presented in tabular images, Figure 12, Figure 13.

4.3.1 Setting for multi-class classification

The following is the train-val-test split for the multi-class classification on four classes of the DeepFakeFace dataset.

- Train set: 80,000 images belonging to 4 classes (20k for each class)
- Validation set: 20,000 images belonging to 4 classes (5k images for each class)
- Test set: 20,000 images belonging to 4 classes (5k images for each class)
- Total: 1,20,000 images of DeepFakeFace dataset.

4.3.2 Setting for cross-generation classification

As there are three different fake classes each generated through a different method, this experiment is to test how the models would perform in a cross-generator classification setting where the classes are from the same dataset, but the generation technique for each of them is different. Out of the three fake classes, one class is kept unseen and is only considered in the test set. The other two fake classes along with the real class is used for training and validation.

Since there are three fake classes, this experiment is repeated thrice, each time having one fake class in test set and the other two for train, val.

As train/val, and test sets therefore, have different classes (names), turned this to a binary classification, where in training/validation, the 2 fake classes will be considered as a fake class (instead of 2 separate classes), and there will be one real class as originally present. Test set has an unseen fake class.

Eg. Test set = Inpainting images -- class name = fake. So, that the test score gives the accuracy of inpainting images classification.

Train, val set = Diffusion v1.5 images, Insightbox images -- class name = fake; IMDB-Wiki -- class name = real

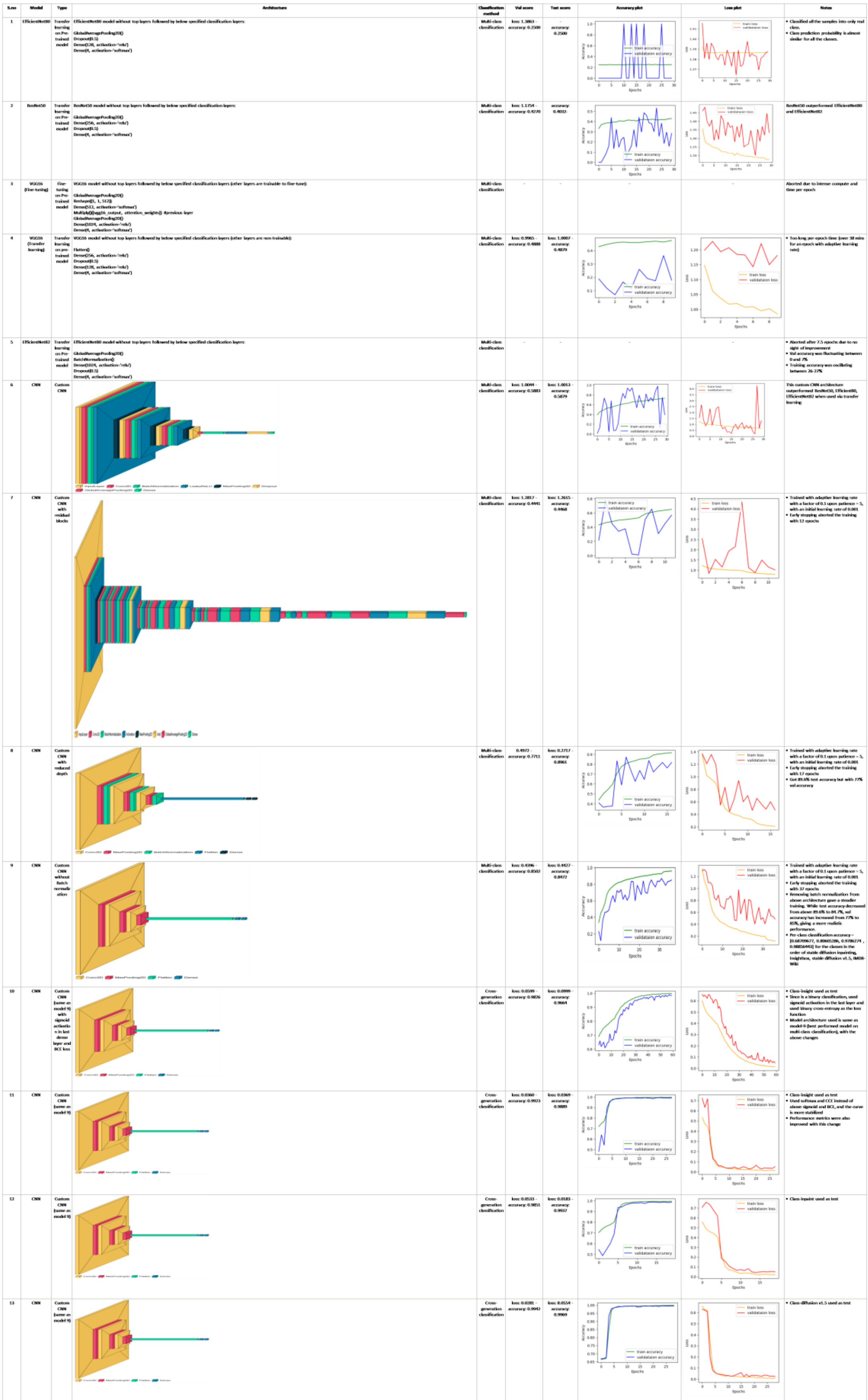
Therefore, two fake classes and the real class for training and validation, and the third fake class is kept for unseen test.

Train set: 72,000 images belonging to 2 classes (24k images for each of 3 train classes; 48k = fake, 24k = real)

Validation set: 18,000 images belonging to 2 classes (6k images for each of 3 classes; 12k = fake, 6k = real)

Test set: 30,000 images belonging to 1 classes (complete set of unseen fake class)

Total: 1,20,000 of DeepFakeFace dataset.



S.no	Model	Type	Architecture	Classification method	Val score	Test score	Accuracy plot	Loss plot	Notes
1	EfficientNetB0	Transfer learning on Pre-trained model	EfficientNetB0 model without top layers followed by below specified classification layers: GlobalAveragePooling2D() Dropout(0.5) Dense(128, activation='relu') Dense(4, activation='softmax')	Multi-class classification	loss: 1.3863 - accuracy: 0.2500	- accuracy: 0.2500			• Classified all the samples into only real class. • Class prediction probability is almost similar for all the classes.
2	ResNet50	Transfer learning on Pre-trained model	ResNet50 model without top layers followed by below specified classification layers: GlobalAveragePooling2D() Dense(256, activation='relu') Dropout(0.5) Dense(4, activation='softmax')	Multi-class classification	loss: 1.1754 - accuracy: 0.4270	accuracy: 0.4032-			ResNet50 outperformed EfficientNetB0 and EfficientNetB2

3	VGG16 (Fine-tuning)	Fine-tuning on Pre-trained model	VGG16 model without top layers followed by below specified classification layers (other layers are trainable to fine-tune):
			
			GlobalAveragePooling2D()
			Reshape((1, 1, 512))
			Dense(512, activation='softmax')
			Multiply()([vgg16_output, attention_weights]) #previous layer
			GlobalAveragePooling2D()
			Dense(1024, activation='relu')
			Dense(4, activation='softmax')
				Multi-class classification	-	-	-	-	Aborted due to intense compute and time per epoch

4	VGG16 (Transfer learning)	Transfer learning on pre-trained model	VGG16 model without top layers followed by below specified classification layers (other layers are non-trainable): Flatten() Dense(256, activation='relu') Dropout(0.5) Dense(128, activation='relu') Dense(4, activation='softmax')	Multi-class classification	loss: 0.9965 - accuracy: 0.4888	loss: 1.0007 - accuracy: 0.4879			• Too long per-epoch time (over 38 mins for an epoch with adaptive learning rate)
5	EfficientNetB2	Transfer learning on Pre-trained model	EfficientNetB0 model without top layers followed by below specified classification layers: GlobalAveragePooling2D() BatchNormalization() Dense(1024, activation='relu') Dropout(0.5) Dense(4, activation='softmax')	Multi-class classification	-	-	-	-	• Aborted after 7.5 epochs due to no sight of improvement • Val accuracy was fluctuating between 0 and 7% • Training accuracy was oscillating between 26-27%
6	CNN	Custom CNN		Multi-class classification	loss: 1.0044 - accuracy: 0.5883	loss: 1.0013 - accuracy: 0.5879			This custom CNN architecture outperformed ResNet50, EfficientB0, EfficientNetB2 when used via transfer learning
7	CNN	Custom CNN with residual blocks		Multi-class classification	loss: 1.2817 - accuracy: 0.4441	loss: 1.2615 - accuracy: 0.4468			• Trained with adaptive learning rate with a factor of 0.1 upon patience = 5, with an initial learning rate of 0.001 • Early stopping aborted the training with 12 epochs
8	CNN	Custom CNN with reduced depth		Multi-class classification	0.4972 - accuracy: 0.7711	loss: 0.2717 - accuracy: 0.8961			• Trained with adaptive learning rate with a factor of 0.1 upon patience = 5, with an initial learning rate of 0.001 • Early stopping aborted the training with 17 epochs • Got 89.6% test accuracy but with 77% val accuracy
9	CNN	Custom CNN without Batch normalization		Multi-class classification	loss: 0.4396 - accuracy: 0.8502	loss: 0.4427 - accuracy: 0.8472			• Trained with adaptive learning rate with a factor of 0.1 upon patience = 5, with an initial learning rate of 0.001 • Early stopping aborted the training with 37 epochs • Removing batch normalization from above architecture gave a steadier training. While test accuracy decreased from above 89.6% to 84.7%, val accuracy has increased from 77% to 85%, giving a more realistic performance. • Per-class classification accuracy - [0.68709677, 0.80605286, 0.9786274 , 0.98856443] for the classes in the order of stable diffusion inpainting, Insightface, stable diffusion v1.5, IMDB-Wiki
10	CNN	Custom CNN (same as model 9) with sigmoid activation in last dense layer and BCE loss		Cross-generation classification	loss: 0.0599 - accuracy: 0.9826	loss: 0.0999 - accuracy: 0.9664			• Class-insight used as test • Since is a binary classification, used sigmoid activation in the last layer and used binary cross-entropy as the loss function • Model architecture used is same as model-9 (best performed model on multi-class classification), with the above changes
11	CNN	Custom CNN (same as model 9)		Cross-generation classification	loss: 0.0360 - accuracy: 0.9923	loss: 0.0369 - accuracy: 0.9889			• Class-insight used as test • Used softmax and CCE instead of above sigmoid and BCE, and the curve is more stabilized • Performance metrics were also improved with this change
12	CNN	Custom CNN (same as model 9)		Cross-generation classification	loss: 0.0533 - accuracy: 0.9851	loss: 0.0183 - accuracy: 0.9937			• Class-inpaint used as test
13	CNN	Custom CNN (same as model 9)		Cross-generation classification	loss: 0.0281 - accuracy: 0.9942	loss: 8.0554 - accuracy: 0.9969			• Class-diffusion v1.5 used as test

Figure 12: Details of the custom CNN, and pretrained models

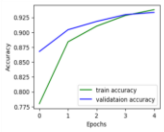
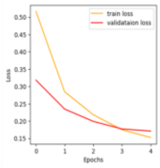
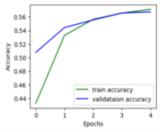
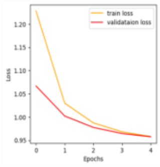
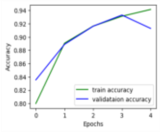
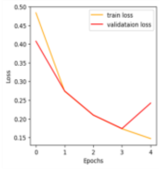
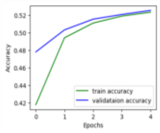
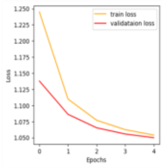
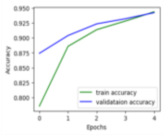
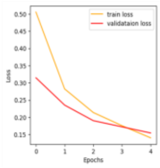
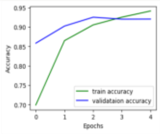
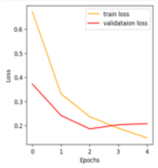
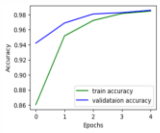
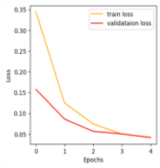
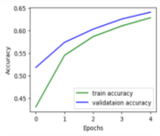
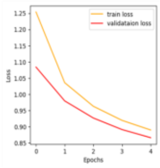
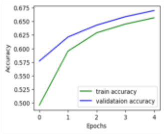
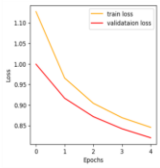
S.no	Model	Attention	Type	Architecture	Trainable Parameters	Classification method	Val score	Test score	Accuracy plot	Loss plot	Execution time (5 epochs)
1	ViT	Self-attention	Fine tuning	vit_tiny_patch16_224	5525188	Multi-class classification	Val Loss: 0.1711, Val Acc: 0.9335	Test Accuracy: 0.9301			320 mins (5.33 hrs)
2	ViT	Self-attention	Transfer learning	vit_tiny_patch16_224	772	Multi-class classification	Val Loss: 0.9577, Val Acc: 0.5674	Test Accuracy: 0.5699			95 mins (1.58 hrs)
3	XCiT – Cross-Covariance Image Transformer	Cross-covariance attention (Transposed version of self-attention that operates across feature channels rather than tokens)	Fine tuning	xcit_nano_12_p16_224	2924740	Multi-class classification	Val Loss: 0.2421, Val Acc: 0.9129	Test Accuracy: 0.9348			225 mins (3.75 hrs)
4	XCiT – Cross-Covariance Image Transformer	Cross-covariance attention	Transfer learning	xcit_nano_12_p16_224	516	Multi-class classification	Val Loss: 1.0499, Val Acc: 0.5258	Test Accuracy: 0.5301			60 mins (1 hr)
5	DeiT	Self-attention Designed for training with limited data using techniques like distillation	Fine tuning	deit_tiny_distilled_patch16_224	5526344	Multi-class classification	Val Loss: 0.1545, Val Acc: 0.9422	Test Accuracy: 0.9429			210 mins (3.5 hrs)
6	Eva02	Multi-scale self-attention (Pyramid of features at different scales combined through aggregation)	Fine tuning	eva02_tiny_patch14_224	5502916	Multi-class classification	Val Loss: 0.2081, Val Acc: 0.9205	Test Accuracy: 0.9276			365 mins (6.08 hrs)
7	CoaT (Co-Scale Conv-Attentional Transformer)	Two components: Co-scale Conv-attentional	Fine tuning	coat_lite_tiny	5402244	Multi-class classification	Val Loss: 0.0414, Val Acc: 0.9858	Test Accuracy: 0.9848			840 mins (14 hrs)
8	ViT with quadrangle attention (Qformer)	Quadrangle attention	Transfer learning	QFormer_transformer_tiny_patch4_window7_224	29145736 (total) 3076 (finetuning)	Multi-class classification	Val Loss: 0.8654, Val Acc: 0.6404	Test Accuracy: 0.6394			468 mins (7.8 hrs)
9	Swin transformer	Shifted-window-based self-attention Hierarchical architecture	Transfer learning	swin_s3_tiny_224	27562750 (total) 3076 (finetuning)	Multi-class classification	Val Loss: 0.8202, Val Acc: 0.6699	Test Accuracy: 0.6625			393 mins (6.55 hrs)

Figure 13: Details of the transformer models trained using finetuning, transfer learning

4.4 Test scenarios

To evaluate and compare the Meta model that was few-shot meta trained, with the other models that were fully trained on the DeepFaceFace dataset, the following two scenarios are considered:

4.4.1 Scenario-1

Test on a completely unseen dataset (OpenForensics-based), where both the Meta model and the other models have never seen this unseen dataset. The comparison was made using three metrics: Accuracy, AUC, and F1 scores. The results are presented in Table 5 and discussed in Section 5.

4.4.2 Scenario-2

To test the meta model and the other models on the test sets of three different datasets namely DGM, DeepFakeFace, and iFakeFaceDB separately. In this case, the other models have never seen the DGM and iFakeFaceDB datasets, nor their test sets. These other models have been exclusively trained on DeepFakeFace dataset as stated in Section 4.3 following the standard training procedure. The Meta model has also never seen the test sets of any of these three datasets, but has been trained on their train sets, through few-shot meta-training, where only a few train samples from one or more classes of these datasets were given for training for a task, based on N-way K-shot selection for that particular task if it includes the corresponding classes in that task. Similar to scenario-1, the comparison was made using the three metrics of accuracy, AUC, and F1 scores, and the results are presented in Table 6 and discussed in Section 5.

4.5 Hardware

The entire dissertation including building and executing the hierarchical multi-agent workflow with RAG and Image synthesis modules, meta-dataset curation, and processing, training, finetuning, transfer learning, all the four categories of 22 models as described in Section 4.3, in multiple settings as described in Sections 4.3.1, and 4.3.2, few-shot meta-model training as described in Section 4.2, and the evaluation of all these models in different settings as described in Section 4.4 are all fully executed on a system having 16 GB of RAM with 8 GB of NVIDIA GeForce RTX 4060 GDDR6 Graphic card with an Intel Core i7 13th gen processor.

5. Results and Observations

5.1 Results of other models – the problem of generalization

Based on the execution results of each of the 22 models shown in Figure 12, and Figure 13, a comparison on various parameters such as on the test accuracy, number of trainable parameters of the model, and execution time is made for the multi-class classification setting.

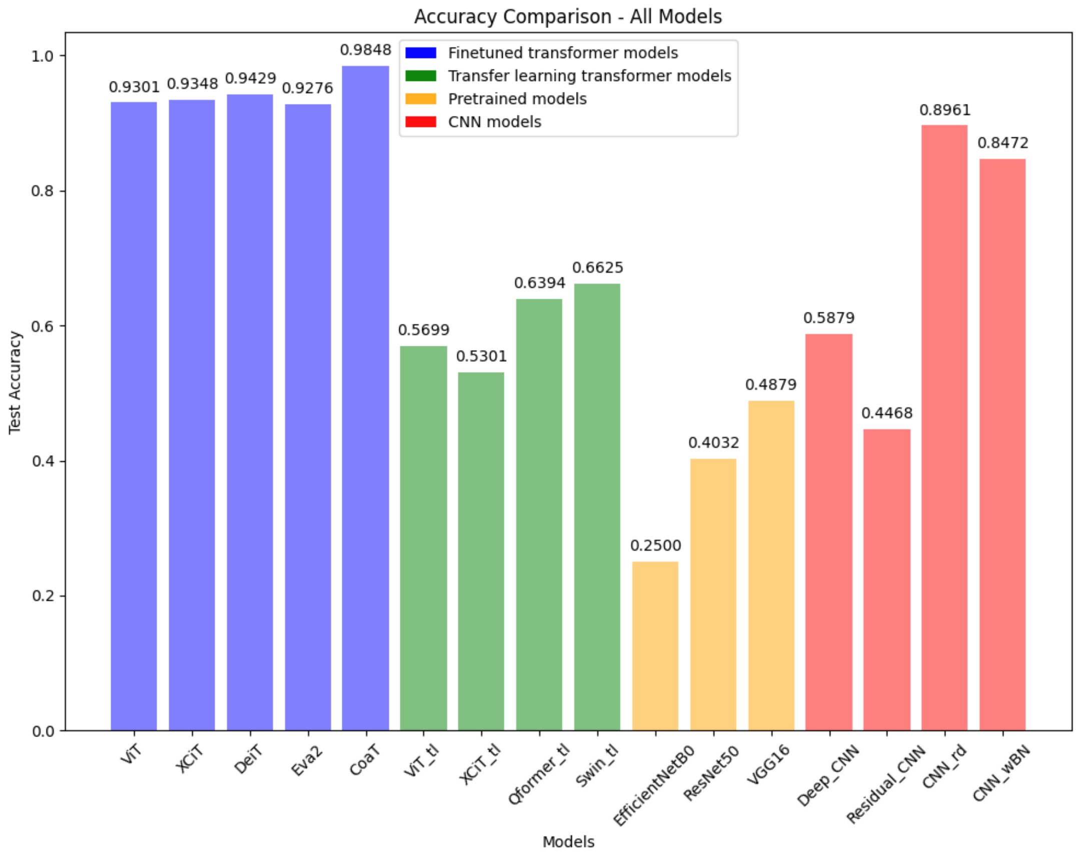


Figure 14: Test accuracy comparison of all the other models (4 categories) trained and tested on DeepFakeFace dataset

Figure 14 depicts the test accuracy performance of all the other models that are trained on the train set of DeepFakeFace dataset, and tested on the test set of the same. It is evident that, fully finetuned transformer models have outperformed all the other three categories of the models tested with a significant margin. Moreover, all these models have reached over a remarkable 92% accuracy and the CoaT model has got the highest with 98.48% test accuracy. Despite not so complex architectures as compared, two of the CNN models have also got a remarkable accuracy of over 84%, with the highest being 89.61% which is just behind the state-of the-art transformers. An important observation made is the extreme drop in performance of the same model architectures when finetuned, and when trained through transfer learning respectively.

There is around a 40% dip in scores. In this case, the worst performance was given by the pre-trained models such as EfficientNetB0, RestNet50, and VGG16 despite their capable architectures.

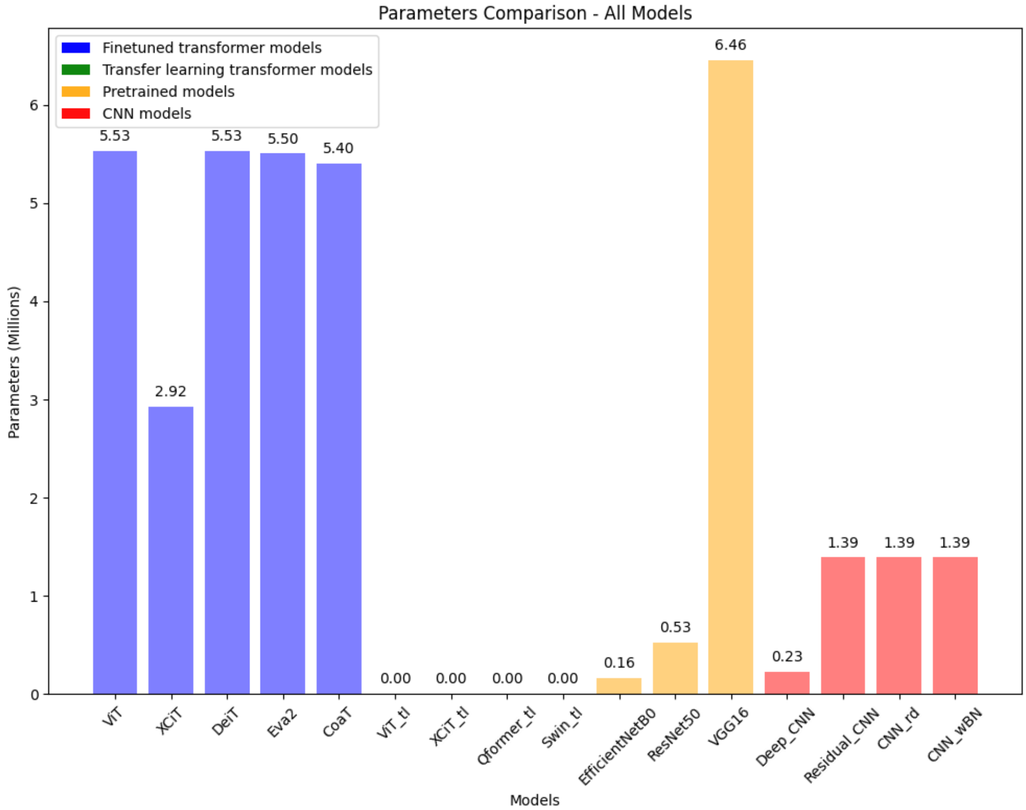


Figure 15: Parameters comparison of all the other models (4 categories) trained and tested on DeepFakeFace dataset

Figure 15 depicts the parameters comparison of all the other models that are trained on the train set of DeepFakeFace dataset, and tested on the test set of the same. From Figure 15, and Figure 16, it is understood that fully finetuning the transformer models based on the model architectures stated in Table 3 takes more execution time due to their large number of trainable parameters, which is the contrast to the CNN models. It is worthy noting that, despite having almost negligible last layer parameters that will be adapted in transfer learning, the transformer models trained through transfer learning have relatively significant and even more execution time than some of the transformer models fully finetuned. Despite having very poor performance, the pretrained models EfficientNetB0, and RestNet50 have taken execution time on par with the other complex models.

Table 3: Model architectures

Model	Architecture
Meta_model	`coat_lite_tiny`
ViT, ViT_tl (transfer learning)	`vit_tiny_patch16_224`
XCiT, XCiT_tl (transfer learning)	`xcit_nano_12_p16_224`
DeiT	`deit_tiny_distilled_patch16_224`
Eva2	`eva02_tiny_patch14_224`
CoaT	`coat_lite_tiny`

Qformer [57]	ViT with quadrangle attention
Swin_tl (transfer learning)	swin_s3_tiny_224
CNN	A 9-layered custom CNN architecture

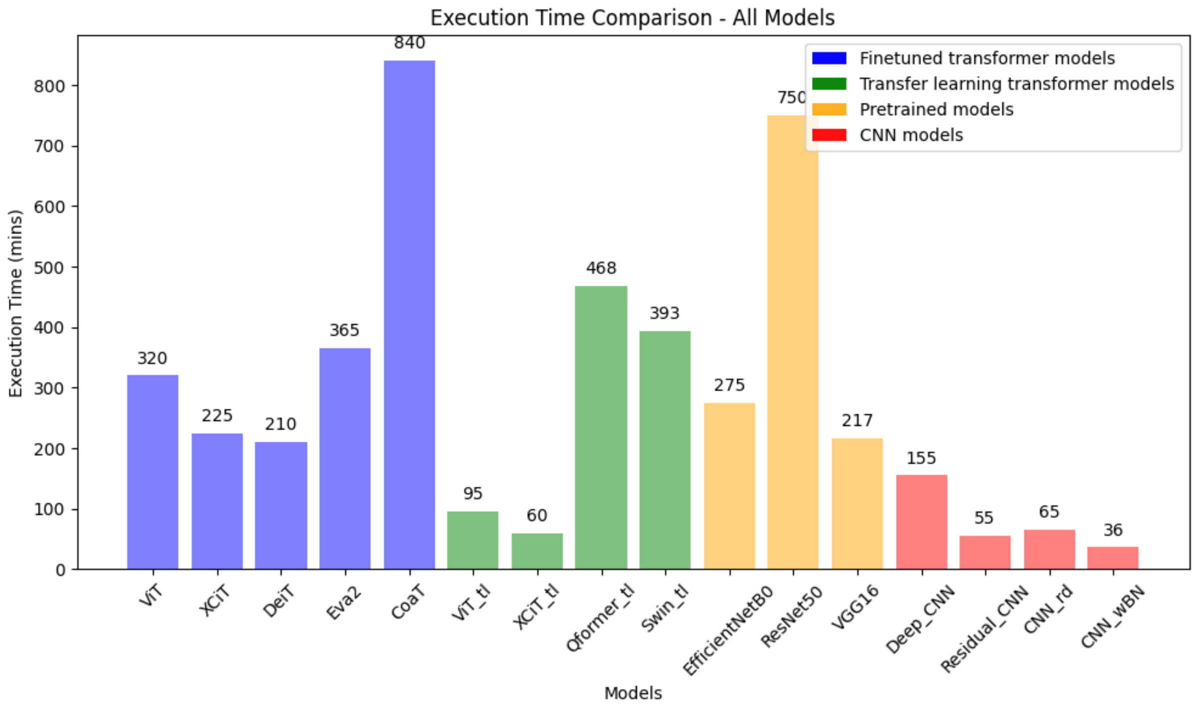


Figure 16: Execution time comparison of all the other models (4 categories) trained and tested on DeepFakeFace dataset

5.2 Meta training stats

The meta-training statistics on the support and query sets are presented in Table 4.

Table 4: Meta-Training Statistics

Meta-Training Stats:	Value
Support Set ACC	0.7641
Support Set Loss	1.632
Query Set ACC	0.7133
Query Set Loss	1.9817

Figure 17, Figure 18 represent the loss and accuracy trends during meta-training for the 30 meta-epochs.

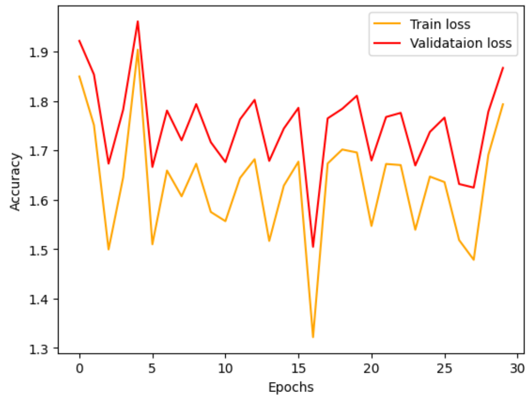


Figure 17: Loss plot of meta-training

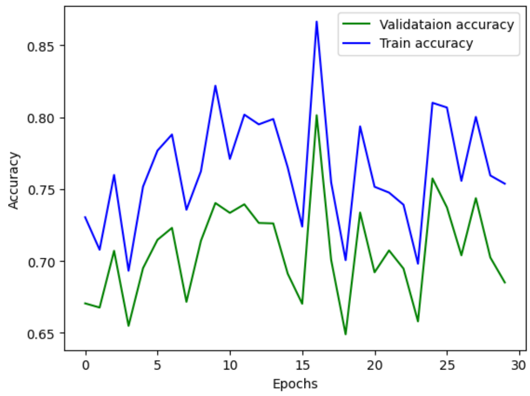


Figure 18: Accuracy plot of meta-training

The accuracy and loss plots have certain fluctuations as the number of meta-epochs though considerable, are relatively less particularly for a meta-training setup where multiple tasks and classes have to be trained upon. Another reason could be the above accuracy and loss graphs denote the accuracy and loss for 300 tasks in a single scale, due to which the actual trends of accuracy and loss for a particular task, or for a particular meta-epoch having 10 different tasks could be lost, as not all tasks get the same performance due to differences in the task complexities. To better understand this, the accuracy plot for each meta-epoch is plotted in Figure 19. In this representation, relatively more stable and smoother curves are observed. In this figure, there are 30 plots, where each plot pertains to a meta-epoch, and each meta-epoch contains 10 diverse tasks. Thus, the accuracy trend noticed in each of the 30 plots here represents overall trend for 10 different tasks. This shows that the curves can get smoother if trained for more meta epochs.

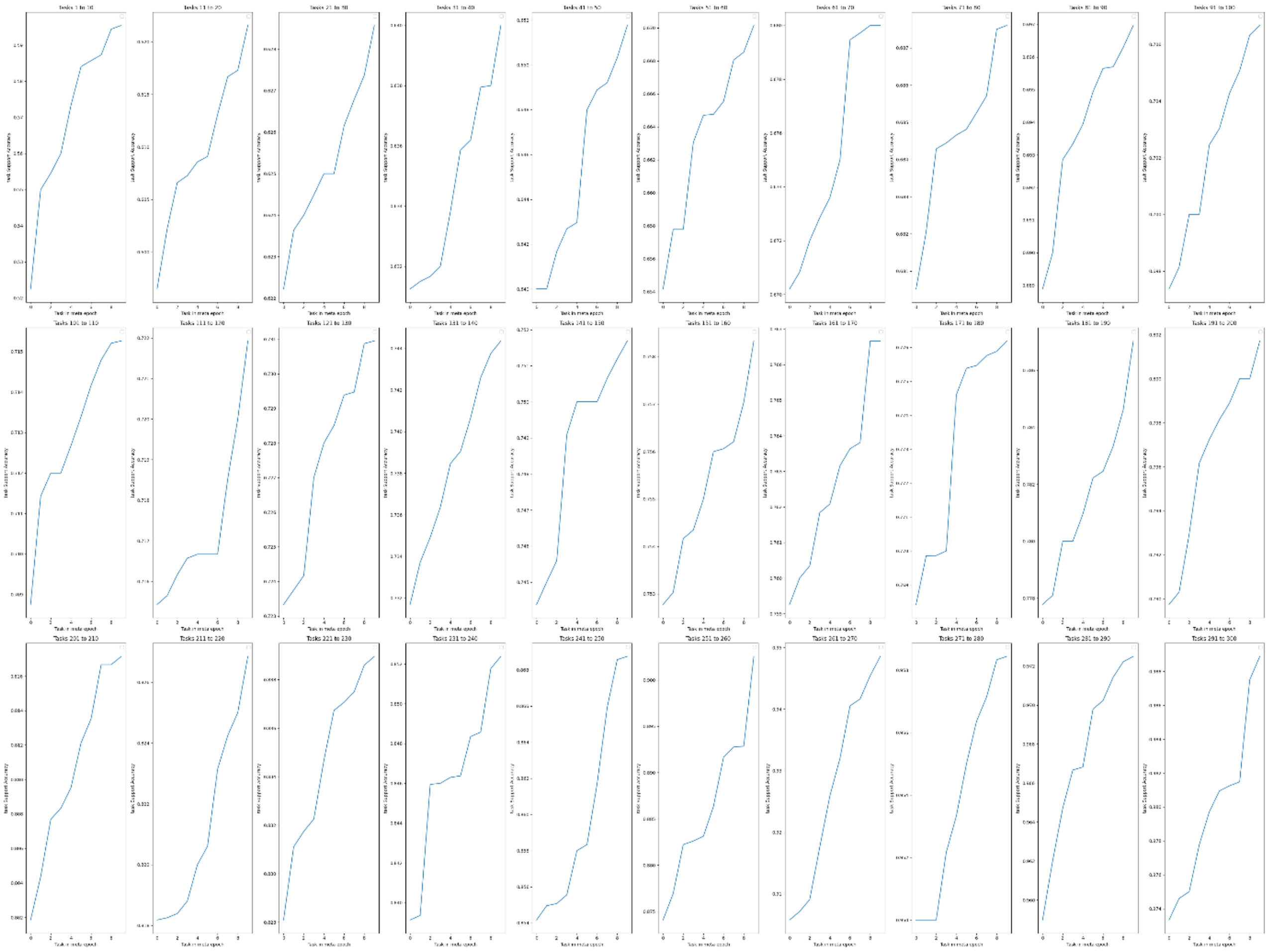


Figure 19: Accuracy trends for every meta-epoch (30), where each epoch pertains to 10 diverse tasks

5.3 Results and discussion on Test scenarios

- For the comparison with the meta model in the different test scenarios considered, the top-5 most performing models amongst the 22 models experimented with as discussed in Section 5.1 are considered. In the first test scenario on the unseen dataset, it is observed that the Meta model has outperformed other fully trained models on all the three metrics. All the other models when tested upon a completely unseen test set, have struggled to reach a random guess performance of 50% accuracy. Figure 20 shows how the performance of the other models has drastically dropped from over 90% to about 50% or less, where the former was when tested on the test set of DeepFakeFace, and the latter was when tested on the unseen OpenForensics-based test set. Additionally, despite the CoaT model, and the Meta model having the same architecture, the CoaT model has got a test accuracy of 46.49% while the Meta model has got 61.51% test accuracy on the unseen test set, showing an improvement of about 15%. However, it is surprising to note that the Swin transformer that was just transfer learned on the DeepFakeFace has got a better performance of 50.7% on the unseen test set than the CoaT model despite having significantly less primary test accuracy, and the Meta model has shown a 10.81% improvement to the best performing other model on the unseen test set.

Table 5: Performance Metrics on Unseen Dataset – OpenForensics-based

Unseen dataset -- OpenForensics-based			
Model	**ACC**	**AUC**	**F1**
Meta_model	0.6151	0.6042	0.6319
ViT	0.4714	0.5197	0.5183
XCiT	0.4495	0.4621	0.4992
DeiT	0.4525	0.4449	0.457
Eva2	0.478	0.4226	0.4633
CoaT	0.4649	0.4675	0.5012

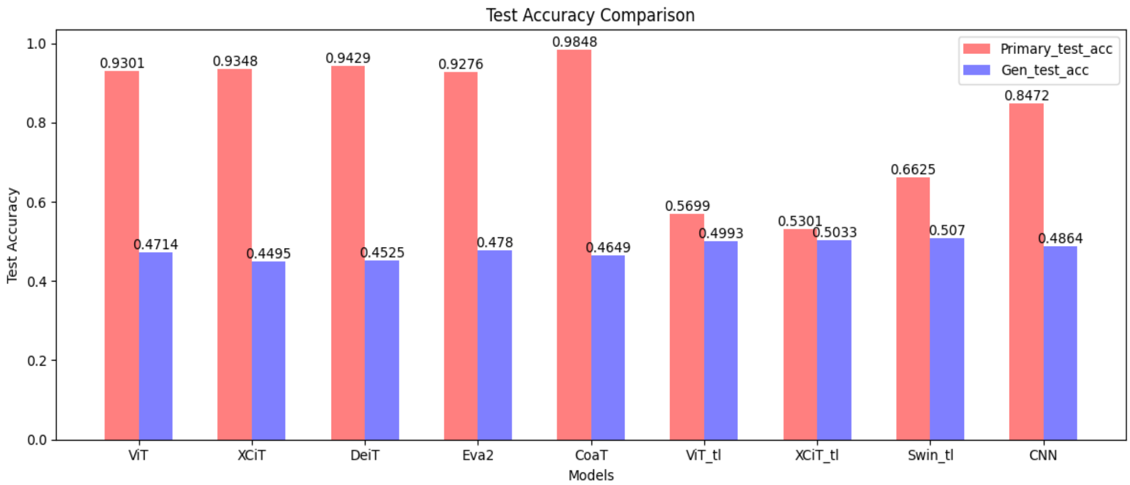


Figure 20: Comparison of test accuracy for different models on the primary test set (DeepFakeface), and on an unseen test set (OpenForensics-based)

- In the second scenario, it is observed that the Meta model remained consistent in the three cases, and has a little improvement of about 3%, 6%, and 7% respectively from the first scenario, as it has been few-shot meta-trained on the train sets each of these datasets. While the other models have significantly outperformed the Meta model in case-2 on the DeepFakeFace upon which they were fully trained, their performance remained similar to the first scenario, with some models having a marginal improvement at cases. However, in the other two cases, the Meta model has outperformed all the other models.

Table 6: Performance Metrics Across Datasets

Dataset	**DGM**			**DeepFakeFace**			**iFakeFaceDB**		
Model	**ACC**	**AUC**	**F1**	**ACC**	**AUC**	**F1**	**ACC**	**AUC**	**F1**
Meta_model	0.6452	0.6879	0.6674	0.6715	0.6938	0.7072	0.6806	0.6726	0.6497
ViT	0.4654	0.4971	0.5633	0.9301	0.9641	0.9831	0.5097	0.4465	0.5318
XCiT	0.4571	0.4526	0.4985	0.9348	0.9219	0.9548	0.4734	0.5135	0.4906

DeiT	0.5139	0.5291	0.5987	0.9429	0.9317	0.9374	0.4895	0.5386	0.5453
Eva2	0.4483	0.4335	0.5142	0.9276	0.9088	0.9519	0.4618	0.4822	0.4782
CoaT	0.4916	0.5437	0.6436	0.9848	0.9904	0.9837	0.5277	0.5897	0.5901

Apparently, the Meta model trained through the proposed algorithm has shown significant improvement in the generalization capabilities across diverse datasets when compared to other similar models. The model was few-shot meta-trained in a restricted infrastructure on a system having 16 GB of RAM with 8 GB of NVIDIA GeForce RTX 4060 GDDR6 Graphic card with an Intel Core i7 13th gen processor. Despite few-shot training in relatively less epochs, the results show that the proposed approach has great potential in enhancing the generalization capabilities of the Meta model which could be realized if it is further trained to the full extent in the described setup with better infrastructure.

5.4 Other observations

- Despite having the same model architecture, and trained on the same dataset, there is a spark difference when finetuned versus when transfer-learned. Adapting only the last or the top layer of the transformer models has not given any better results, and in cases, even got underperformed by simple CNN models. While finetuning the models fully has revealed the true potential of the transformer models, where all of them have got top-notch performances. This is represented in Figure 21.

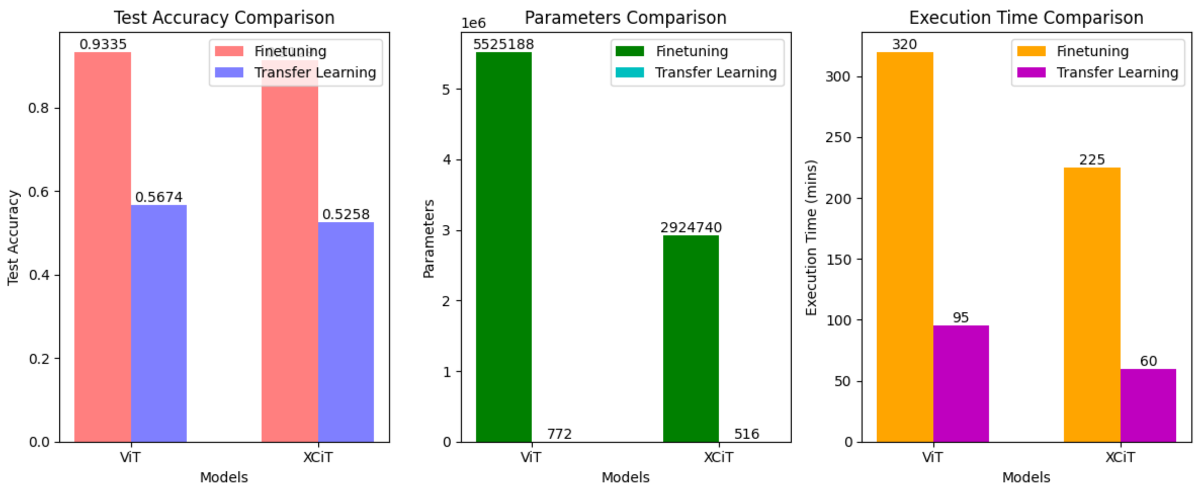


Figure 21: Transfer-learning vs Finetuning transformer models

- From the performance vs trainable parameters vs execution time tradeoff graph as represented in Figure 22, XCiT model has a better balance amongst the other models, and does not compromise much on the performance, yet with almost half the parameters of other transformer models. Due to this, the execution time is also far less, making it a model with better tradeoff specifically in a resource constraint setup.

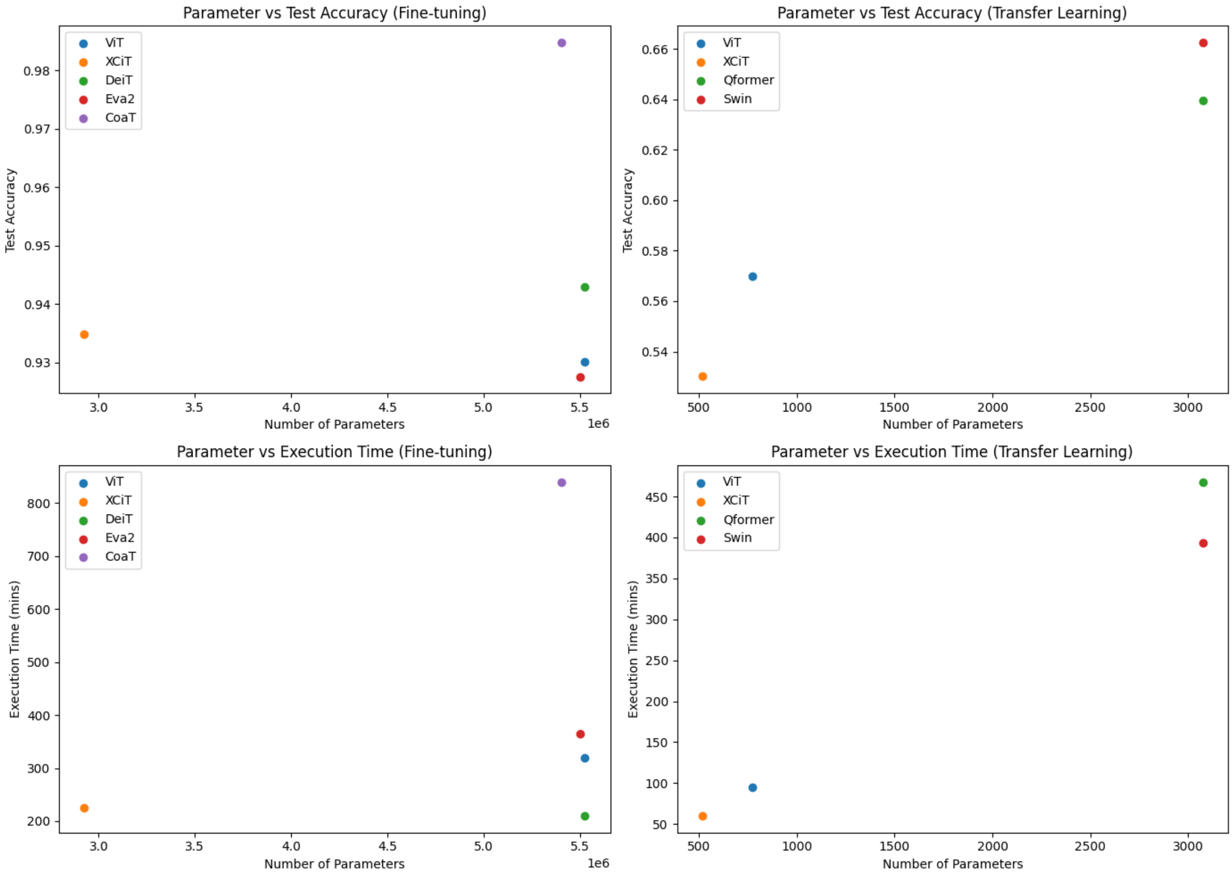


Figure 22: Parameters vs Test accuracy vs Execution time tradeoff

- It is observed that an increase in the number of classes per task is detrimental to query set accuracy, while increase in the number of samples per class in a task has an overall positive effect on the query set accuracy. This is represented in Figure 23, Figure 24 respectively.

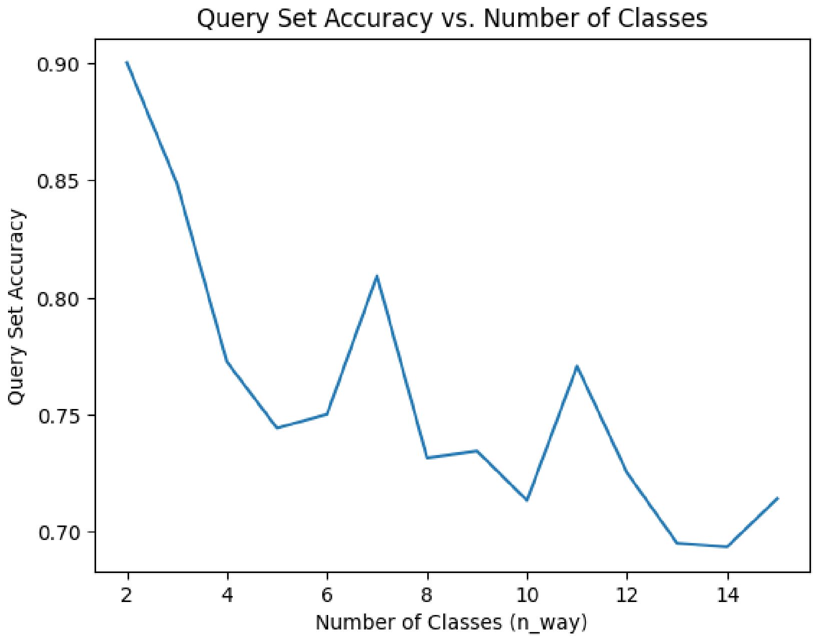


Figure 23: N-Way vs Query Set Accuracy

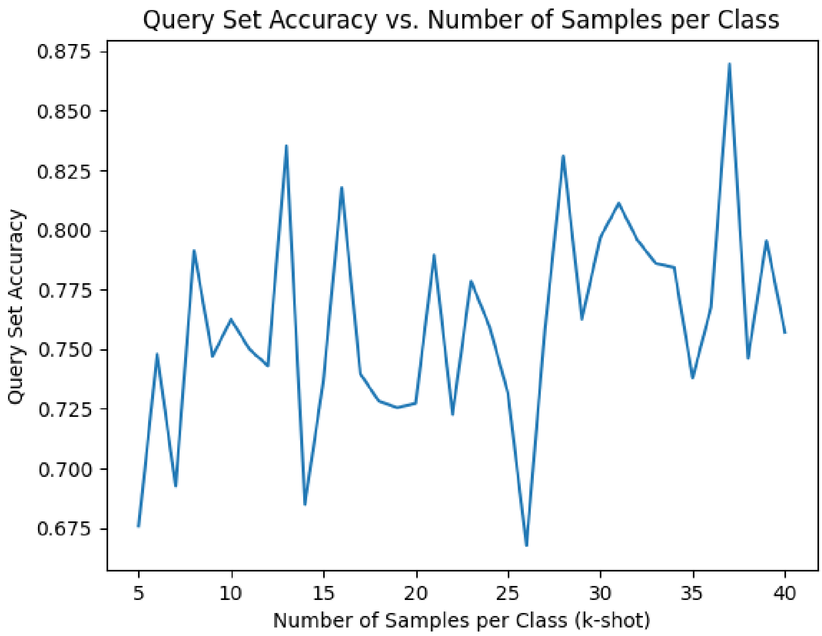


Figure 24: K-Shot vs Query Set Accuracy

6. Conclusion and Future directions

6.1 Conclusion

This work deals with the crucial challenges of model generalization, adversarial robustness, and adaptation to dynamic data drift. The problem is setup in the domain of deepfake detection. However, the considered three precise challenges go beyond deepfake detection and hold utmost importance for any classifier in any other domain or setup.

A holistic framework with two major modules of adversarial meta-training, and hierarchical multi-agent workflow with RAG and custom image synthesis modules is proposed, designed, implemented, and experimented in this work. The proposed meta-algorithm identifies the weaknesses and strengths of the model by ranking the support set samples using the proposed metric $M_{adaptive}$ which ranks the samples in the order of Wrong predictions with large margin > Wrong predictions with small margin > Right predictions with small margin > Right predictions with large margin. This helps to identify those sets of samples where the model is struggling to make a decision, and where the model is confident enough. Corresponding ensemble augmented and adversarially perturbed samples are generated and added to the support set of each of the respective tasks, in every inner step, there by making the model fix its weaknesses thus, enhancing its detection capabilities and generalization, and enhance its strengths and confidence making them adversarially robust. This way, the proposed meta-learning algorithm enhances the generalization and robustness of the model, which is demonstrated in the experimental test scenarios in this dissertation where the meta model trained has shown a consistent performance across different datasets. Additionally, the proposed multi-agent workflow with RAG and custom sample synthesis modules helps in collecting real-time information from diverse sources including web and news outlets, research papers and technical articles, web-based search, apart from the locally curated knowledge base. This information is processed by a hierarchy of agents, in doing various relevant tasks as described in this thesis for attack pattern generation, and few-shot prompt generation, which will be subsequently used by the sample synthesis module to generate the corresponding few-shot samples pertaining to the attack pattern given or generated. The constant addition of these generated samples to the support set of tasks, makes the model realize the emerging and unforeseen trends, making it quickly adapt to newer attacks and scenarios. The dissertation elaborates on the working of each of the above-described modules along with the discussing their outputs and results.

6.2 Future scope of work

As part of the future scope of work, the following could be some directions to begin with to sophisticate the proposed solution:

Multi-modal RAG system with Semantic chunking can be integrated into the current RAG design to get information from multi-modal inputs and more contextually relevant chunks can therefore, be made. The agents could be sophisticated by adding further agents such as for content moderation to eliminate any concerning information from processing. Besides, the agents can be better adapted to their very precise tasks assigned using instruction tuning by curating corresponding instruction datasets. The performance of the model can be further improved if a bigger architecture or variant is taken instead of the lite versions. Additionally, training them for more epochs, also, if possible, on the complete meta-dataset instead of few-shot training can all lead to improved performance of the meta-model. The current implementation primarily focusses on the image deepfakes, but its design holds true for other modalities as well. As part of the future work, it can be extended to other data modalities, making it a multi-modal solution increasing its spectrum of usecases and applications.

References

[1] S. Suratkar and F. Kazi, “Deep fake video detection using transfer learning approach,” Arabian journal for science and engineering, vol. 48, pp. 1–11, 10 2022.

[2] M. Kim, S. Tariq, and S. S. Woo, “Fretal: Generalizing deepfake detection using knowledge distillation and representation learning,” in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1001–1012, 2021.

[3] T. Kim, J. Choi, H. Cho, H. Lim, and J. Choi, “Domain generalization for face forgery detection by style transfer,” in 2024 IEEE International Conference on Consumer Electronics (ICCE), pp. 1–5, 2024.

[4] H. Song, S. Huang, Y. Dong, and W.-W. Tu, “Robustness and generalizability of deepfake detection: A study with diffusion models,” 2023.

[5] B. Li, J. Sun, C. M. Poskitt, and X. Wang, “How generalizable are deepfake image detectors? an empirical study,” 2024.

[6] L. A. Passos, D. Jodas, K. A. P. Costa, L. A. Souza J´unior, D. Rodrigues, J. Del Ser, D. Camacho, and J. P. Papa, “A review of deep learning-based approaches for deepfake content detection,” Expert Systems, vol. 41, Feb. 2024.

[7] S. Kamat, S. Agarwal, T. Darrell, and A. Rohrbach, “Revisiting generalizability in deepfake detection: Improving metrics and stabilizing transfer,” in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pp. 426–435, October 2023.

[8] M. Ivanovska and V. Struc, “On the vulnerability of deepfake detectors to attacks generated by denoising diffusion models,” in 2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW), (Los Alamitos, CA, USA), pp. 1051–1060, IEEE Computer Society, jan 2024.

[9] N. Carlini and H. Farid, “Evading deepfake-image detectors with white- and black-box attacks,” in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2804–2813, 2020.

[10] Y. Hou, Q. Guo, Y. Huang, X. Xie, L. Ma, and J. Zhao, “Evading deepfake detectors via adversarial statistical consistency,” 2023.

[11] A. Devasthale and S. Sural, “Adversarially robust deepfake video detection,” in 2022 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 396–403, 2022.

[12] A. Hooda, N. Mangaokar, R. Feng, K. Fawaz, S. Jha, and A. Prakash, “D4: Detection of adversarial diffusion deepfakes using disjoint ensembles,” 2023.

[13] S. Khan, J.-C. Chen, W.-H. Liao, and C.-S. Chen, “Adversarially robust deepfake detection via adversarial feature similarity learning,” in MultiMedia Modeling: 30th International Conference, MMM 2024, Amsterdam, The Netherlands, January 29 – February 2, 2024, Proceedings, Part III, (Berlin, Heidelberg), p. 503–516, Springer-Verlag, 2024.

[14] P. Neekhara, B. Dolhansky, J. Bitton, and C. C. Ferrer, “Adversarial threats to deepfake detection: A practical perspective,” in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 923–932, 2021.

[15] C. Galdi, M. Panariello, M. Todisco, and N. Evans, “2d-malafide: Adversarial attacks against face deepfake detection systems,” 2024.

[16] N. Lim, M. Y. Kuan, M. Pu, M. Lim, and C. Y. Chong, “Metamorphic testing-based adversarial attack to fool deepfake detectors,” in 2022 26th International Conference on Pattern Recognition (ICPR), (Los Alamitos, CA, USA), pp. 2503–2509, IEEE Computer Society, aug 2022.

[17] J. Kim, T. Kim, J. Kim, and S. S. Woo, “Evading deepfake detectors via high quality face pre-processing methods,” in 2022 26th International Conference on Pattern Recognition (ICPR), (Los Alamitos, CA, USA), pp. 1937–1944, IEEE Computer Society, aug 2022.

[18] C. Liu, H. Chen, T. Zhu, J. Zhang, and W. Zhou, “Making deepfakes more spurious: Evading deep face forgery detection via trace removal attack,” IEEE Transactions on Dependable and Secure Computing, vol. 20, pp. 5182–5196, nov 2023.

[19] F. Tassone, L. Maiano, and I. Amerini, “Continuous fake media detection: Adapting deepfake detectors to new generative techniques,” Computer Vision and Image Understanding, vol. 249, p. 104143, 2024.

[20] A. Nichol, J. Achiam, and J. Schulman, “On first-order meta-learning algorithms,” 2018.

[21] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen, “Deep learning for deepfakes creation and detection: A survey,” Computer Vision and Image Understanding, vol. 223, p. 103525, Oct. 2022.

[22] Y. Mirsky and W. Lee, “The creation and detection of deepfakes: A survey,” ACM Computing Surveys, vol. 54, p. 1–41, Jan. 2021.

[23] Z. Akhtar, “Deepfakes generation and detection: A short survey,” Journal of Imaging, vol. 9, no. 1, 2023.

[24] S. Mittal, K. Thakral, P. Majumdar, M. Vatsa, and R. Singh, “Are face detection models biased?,” 2022.

[25] K. Narayan, H. Agarwal, K. Thakral, S. Mittal, M. Vatsa, and R. Singh, “Df-platter: Multi-face heterogeneous deepfake dataset,” in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9739–9748, 2023.

[26] K. Narayan, H. Agarwal, K. Thakral, S. Mittal, M. Vatsa, and R. Singh, “Deephy: On deepfake phylogeny,” 2022.

[27] R. Shao, T. Wu, J. Wu, L. Nie, and Z. Liu, “Detecting and grounding multi-modal media manipulation and beyond,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 46, no. 8, pp. 5556–5574, 2024.

[28] S. Wang, Y. Zhu, R. Wang, A. Dharmasiri, O. Russakovsky, and Y. Wu, “Deter: Detecting edited regions for deterring generative manipulations,” 2023.

[29] Y. Wang, Z. Huang, Z. Ma, and X. Hong, “Linguistic profiling of deepfakes: An open database for next-generation deepfake detection,” ArXiv, vol. abs/2401.02335, 2024.

[30] S. Jia, X. Li, and S. Lyu, “Model attribution of face-swap deepfake videos,” in 2022 IEEE International Conference on Image Processing (ICIP), pp. 2356–2360, 2022.

[31] S. Jia, M. Huang, Z. Zhou, Y. Ju, J. Cai, and S. Lyu, “Au-

tosplice: A text-prompt manipulated image dataset for media forensics,” in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 893–903, 2023.

[32] P. Korshunov and S. Marcel, “Deepfakes: a new threat to face recognition? assessment and detection,” ArXiv, vol. abs/1812.08685, 2018.

[33] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, “On the detection of digital face manipulation,” in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), (Los Alamitos, CA, USA), pp. 5780–5789, IEEE Computer Society, jun 2020.

[34] J. C. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, H. Proenc¸a, and J. Fierrez, “Ganprintr: Improved fakes and evaluation of the state of the art in face manipulation detection,” IEEE Journal of Selected Topics in Signal Processing, vol. 14, no. 5, pp. 1038–1048, 2020.

[35] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df: A large-scale challenging dataset for deepfake forensics,” 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3204–3213, 2019.

[36] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, “Faceforensics++: Learning to detect manipulated facial images,” in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), (Los Alamitos, CA, USA), pp. 1–11, IEEE Computer Society, nov 2019.

[37] T. Zhou, W. Wang, Z. Liang, and J. Shen, “Face forensics in the wild,” in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5774–5784, 2021.

[38] B. Peng, H. Fan, W. Wang, J. Dong, Y. Li, S. Lyu, Q. Li, Z. Sun, H. Chen, B. Chen, Y. Hu, S. Luo, J. Huang, Y. Yao, B. Liu, H. Ling, G. Zhang, Z. Xu, C. Miao, C. Lu, S. He, X. Wu, and W. Zhuang, “Dfgc 2021: A deepfake game competition,” in 2021 IEEE International Joint Conference on Biometrics (IJCB), pp. 1–8, 2021.

[39] B. Zi, M. Chang, J. Chen, X. Ma, and Y.-G. Jiang, “Wilddeepfake: A challenging real-world dataset for deepfake detection,” Proceedings of the 28th ACM International Conference on Multimedia, 2020.

[40] X. Yang, Y. Li, and S. Lyu, “Exposing deep fakes using inconsistent head poses,” in ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8261–8265, 2019.

[41] A. KoC¸ ak and M. Alkan, “Deepfake generation, detection and datasets: a rapid-review,” in 2022 15th International Conference on Information Security and Cryptography (ISC-TURKEY), pp. 86–91, 2022.

[42] R. Purohit, Y. Sane, D. Vaishampayan, S. Vedantam, and M. Singh, “Ai vs. human vision: A comparative analysis for distinguishing ai-generated and natural images,” in 2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), pp. 1–7, 2024.

[43] J. Bobulski and M. Kubanek, Detection of Fake Facial Images and Changes in Real Facial Images, pp. 110–122. Computational Collective Intelligence, 08 2024.

[44] S. Waseem, S. A. R. S. Abu Bakar, B. A. Ahmed, Z. Omar, T. A. E. Eisa, and M. E. E. Dalam, "Deepfake on face and expression swap: A review," IEEE Access, vol. 11, pp. 117865–117906, 2023.

[45] A. Malik, M. Kuribayashi, S. M. Abdullahi, and A. N. Khan, "Deepfake detection for human face images and videos: A survey," IEEE Access, vol. 10, pp. 18757–18775, 2022.

[46] R. Chauhan, R. Popli, and I. Kansal, "A systematic review on fake image creation techniques," in 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), pp. 779–783, 2023.

[47] M. M. Diniz and A. Rocha, "Open-set deepfake detection to fight the unknown," in ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 13091–13095, 2024.

[48] J. Liu, M. Zhang, J. Ke, and L. Wang, "Advshadow: Evading deepfake detection via adversarial shadow attack," in ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4640–4644, 2024.

[49] N. Carlini and H. Farid, "Evading deepfake-image detectors with white- and black-box attacks," in IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2804–2813, 06 2020.

[50] S. Abdullah, A. Cheruvu, S. Kanchi, T. Chung, P. Gao, M. Jadliwala, and B. Viswanath, "An analysis of recent advances in deepfake image detection in an evolving threat landscape," in 2024 IEEE Symposium on Security and Privacy (SP), (Los Alamitos, CA, USA), pp. 91–109, IEEE Computer Society, may 2024.

[51] A. Naitali, M. Ridouani, F. Salahdine, and N. Kaabouch, "Deepfake attacks: Generation, detection, datasets, challenges, and research directions," Comput., vol. 12, p. 216, 2023.

[52] R. Shao, T. Wu, and Z. Liu, "Detecting and grounding multi-modal media manipulation," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[53] J. C. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, H. Proenc¸a, and J. Fierrez, "GANprintR: Improved Fakes and Evaluation of the State-of-the-Art in Face Manipulation Detection," 2019.

[54] F. Guillaro, D. Cozzolino, A. Sud, N. Dufour, and L. Verdoliva, "Trufor: Leveraging all-round clues for trustworthy image forgery detection and localization," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 20606–20615, June 2023.

[55] Z. Yan, Y. Zhang, X. Yuan, S. Lyu, and B. Wu, "Deepfakebench: A comprehensive benchmark of deepfake detection," in Advances in Neural Information Processing Systems (A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds.), vol. 36, pp. 4534–4565, Curran Associates, Inc., 2023.

[56] T.-N. Le, H. H. Nguyen, J. Yamagishi, and I. Echizen, "Openforensics: Large-scale challenging dataset for multi-face forgery detection and segmentation in-the-wild," in International Conference on Computer Vision, 2021.

[57] Q. Zhang, J. Zhang, Y. Xu and D. Tao, "Vision Transformer With Quadrangle Attention," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3608-3624, May 2024, doi: 10.1109/TPAMI.2023.3347693.